Calabazas Creek Research Inc.

Calabazas Creek Research Inc.
690 Port Drive
San Mateo, CA 94404

Phone 650 312-9575
Fax 650 312-9536

BOA, Beam Optics Analyzer
A Particle-In-Cell Code

DE-FG02-03ER83616

Program Summary

The program was tasked with implementing time dependent analysis of charges particles into an existing finite element code with adaptive meshing, called Beam Optics Analyzer (BOA). BOA was initially funded by a DOE Phase II program to use the finite element method with adaptive meshing to track particles in unstructured meshes. It uses modern programming techniques, state-of-the-art data structures, so that new methods, features and capabilities are easily added and maintained. This Phase II program was funded to implement plasma simulations in BOA and extend its capabilities to model thermal electrons, secondary emissions, self magnetic field and implement a more comprehensive post-processing and feature-rich GUI.

The program was successful in implementing thermal electrons, secondary emissions, and self magnetic field calculations. The BOA GUI was also upgraded significantly, and CCR is receiving interest from the microwave tube and semiconductor equipment industry for the code. Implementation of PIC analysis was partially successful. Computational resource requirements for modeling more than 2000 particles begin to exceed the capability of most readily available computers. Modern plasma analysis typically requires modeling of approximately 2 million particles or more. The problem is that tracking many particles in an unstructured mesh that is adapting becomes inefficient. In particular memory requirements become excessive. This probably makes particle tracking in unstructured meshes currently unfeasible with commonly available computer resources. Consequently, Calabazas Creek Research, Inc. is exploring hybrid codes where the electromagnetic fields are solved on the unstructured, adaptive mesh while particles are tracked on a fixed mesh. Efficient interpolation routines should be able to transfer information between nodes of the two meshes. If successfully developed, this could provide high accuracy and reasonable computational efficiency.

FINAL REPORT

BOA, Beam Optics Analyzer
A Particle-In-Cell Code

Principal Investigator: Thuc Bui

Grant Number DE-FG02-03ER83616

November 2007

Contents

Abstract

This final report summarizes the theoretical background for implementing a finite element, Particle-in-Cell (PIC) code and describes the pertinent algorithms used successfully by Beam Optics Analyzer (BOA). BOA is a 3D, finite element analysis program with fully automatic and adaptive meshing that can simulate plasmas and trace particle in complex geometries. A major effort was development of the graphical user interface. Its powerful post-processing capability and ease of use will be demonstrated via some practical examples.

Introduction

Beam Optics Analyzer (BOA) was initially funded by a DOE Phase II program to use the finite element method with adaptive meshing to track particles in unstructured meshes. It uses modern programming techniques, state-of-the-art data structures, so that new methods, features and capabilities are easily added and maintained [4, 5]. It can model thermionic emission and injected particles from prescribed emitting surfaces. It imports the 3D geometries in either ACIS or Parasolid format from most commercial CAD packages. A graphical user interface (GUI) is used to assign boundary conditions, material properties and other pertinent analysis parameters. The initial GUI post-processing capabilities provided simple plotting of scalar potential contours, electric field and particle trajectories. It could import the magnetic fields from various packages such as Maxwell2D, Maxwell3D, Pandira or 1D axial field to focus the electron beams. With its fully automatic and adaptive meshing, the code successfully modeled and benchmarked several electron guns and collectors. However, the code lacked the ability to model secondary emission, self magnetic field and plasmas, for which proper prescribing of the initial conditions by smooth loading of particles (both electrons and ions) for quiet starts is a must. More importantly, the requirements of the ordinary differential equation (ODE) integrator used by the particle pusher for particle tracing are different from those of plasma simulations. In particle tracing, particles are not required to be synchronized in time steps. Thus, much higher order integrator can be used as long as it meets the particle Courant condition whose length parameter is the size of the mesh on the particle path. In plasma simulations, all particles must be pushed synchronically. A different ODE integrator is required. A very high order accuracy integrator is not needed, but a very efficient one is preferred, since the number of particles in plasma simulations can be quite large.

This Phase II program was funded to implement plasma simulations in BOA and extend its capabilities to model thermal electrons, secondary emissions, self magnetic field and implement a more comprehensive post-processing and feature-rich GUI [22, 23]. This

report will first summarize the code capabilities with a problem statement followed by their integral formulations and technical issues that are computationally solved. The next section demonstrates set up and post process solution fields and particles in some examples accompanied by detailed visualization of the results. Finally, summary remarks will be provided.

Problem Statement

1. Electrostatics

BOA uses the quasi-static model or frozen field approximation for plasma simulations. The electric and magnetic fields are not coupled, but are solved separately, but they must be solved at the same time that the charge and current densities are specified. The frozen electric E field is obtained by solving Poisson's equation for the scalar potential Φ in the problem domain Ω

$$\nabla \cdot \varepsilon(\mathbf{x}) \nabla \Phi(\mathbf{x},t) = -\rho(\mathbf{x},t) \qquad \text{on } \Omega \tag{1}$$

to satisfy the prescribed boundary conditions

$$\Phi = V_0 \qquad \text{on } \Gamma_d \tag{2}$$

$$-\mathbf{D} \cdot \mathbf{n} = \rho_{s0} \qquad \text{on } \Gamma_n \tag{3}$$

where the electric field

$$\mathbf{E} \equiv -\nabla \Phi \tag{4}$$

and the displacement current

$$\mathbf{D} = \varepsilon \mathbf{E} \tag{5}$$

ε is the permittivity, ρ the charge density field and Γ_d and Γ_n are Dirichlet and Neumann surfaces respectively. The surface charge ρ_{s0} is prescribed on the Neumann boundary. Across the interfaces of regions of different materials, the following continuity condition must also be met

$$(\mathbf{D}_1 - \mathbf{D}_2) \cdot \mathbf{n} = \rho_s \tag{6}$$

2. Magnetostatics

The frozen magnetic flux density B field is derived from the vector potential **A** in the problem domain Ω, which is the solution of the Curl-Curl equation

$$\nabla \times \frac{1}{\mu} \nabla \times \mathbf{A} = \mathbf{J} \qquad \text{on } \Omega \tag{7}$$

with the prescribed boundary conditions

$$\mathbf{B} \cdot \mathbf{n} = B_{0n} \qquad \text{on } \Gamma_d \tag{8}$$

$$\mathbf{n} \times \mathbf{H} = \mathbf{J}_{s0} \qquad \text{on } \Gamma_n \tag{9}$$

where

$$\mathbf{B} \equiv \nabla \times \mathbf{A} \tag{10}$$

and

$$\mathbf{B} = \mu \mathbf{H} \tag{11}$$

Equations (8) and (9) can be rewritten in term of the vector of potential as follows

$$\mathbf{n} \times \mathbf{A} = \mathbf{A}_{0t} \qquad \text{on } \Gamma_d \tag{12}$$

$$\mathbf{n} \times \frac{1}{\mu} \nabla \mathbf{A} = \mathbf{J}_{s0} \qquad \text{on } \Gamma_n \tag{13}$$

μ is the permeability, $\mathbf{J}_{s0}$ the surface current on Neumann surfaces and **J** the current density vector, which is due to the vector property of Eq. (7) that **J** must be divergence-free i.e.,

$$\nabla \cdot \mathbf{J} = 0 \tag{14}$$

The continuity conditions across the interfaces of regions of different materials are

$$\mathbf{n} \times \left(\mathbf{A}_1 - \mathbf{A}_2 \right) = \mathbf{0} \tag{15}$$

$$\mathbf{n} \times \left(\mathbf{H}_1 - \mathbf{H}_2 \right) = \mathbf{0} \tag{16}$$

Solution of the curl-curl equation (16) involves a non-trivial space of the gradient of a scalar because substituting the vector potential by

$$\mathbf{A} = \mathbf{A}_0 + \nabla \Psi \tag{17}$$

and using the vector identity $\nabla \cdot \nabla \times \mathbf{A} = 0$, we can see that $\mathbf{A}_0$ is the solution we are seeking, but there is the nullspace $\nabla \Psi$. To fully and uniquely determine a vector field requires specifying both its curl and its divergence as stated by a fundamental theorem by Helmholtz. Even though Eq. (7) specifies the curl of **A**, its divergence is not yet

determined. Thus, to have a unique solution of a vector field is to assign a value to the divergence of **A**. This value may be selected as will without affecting the physical problem, for all possible values still yield the same **B**. Different choices for divergence of **A** are referred to as choices of gauge. The best-known and most common gauge in magnetostatics is the Coulomb gauge, which is

$$\nabla \cdot \mathbf{A} = 0 \tag{18}$$

BOA does not gauge the vector potential at all and solves the curl-curl equation directly. Using the curl-curl equation for magnetostatics in regions that have current sources will be more natural for inclusion of non-uniform materials. More importantly, the same technology developed for this type of equation is also applicable to the full set of Maxwell's equations when a time-domain solver is needed. However, the curl-curl operator has a large nullspace, and without gauging, solutions to Eq. (7) will not converge without extra effort. Earlier works [26, 27] attempted to eliminate this nullspace, consequently making the matrix generated from the discretization of the **curl-curl** operator nonsingular to improve the matrix solution. Unfortunately, this technique, based on the network (graph) theory to achieve the matrix nonsingularity, is complex and difficult to implement. It turns out that using the conjugate-gradient method to solve the singular matrix, the solution converges very efficiently with a proper conditioning of the source term making it divergence-free [26, 28, 29]. Techniques to enforce the compatibility of the curl-curl equation and insure the current density is divergence-free will be described in detail in the next section.

3. Newton-Lorentz Equation of Motion

We can obtain the particle velocity and position by applying the electric field and magnetic flux density and solving the above electrostatic and magnetostatic equations for the Newton-Lorentz equation of motion, which is

$$\frac{d\mathbf{u}}{dt} = \eta\left(\mathbf{E} + \mathbf{v} \times \mathbf{B}\right) \tag{19}$$

and

$$\frac{d\mathbf{x}}{dt} = \mathbf{v} \tag{20}$$

where $\eta = q/m$, $\mathbf{u} = \gamma\mathbf{v}$, and

$$\gamma = \sqrt{\frac{1}{1-\left(v/c\right)^2}} = \sqrt{1+\left(u/c\right)^2} \tag{21}$$

Solutions to Equations (19) and (20) for **u** and **x** will require the initial conditions for each launched particle.

Equations (1) to (6) are the *strong* statements of Electrostatics, and Eqs. (7) to (16) are those for the Magnetostatics. Unlike some other approximation methods, notably the finite difference method, we begin directly with the strong statement. The finite element field solvers in BOA required a reformulation, since they did not work directly with the *strong* forms ,but instead with their integral formulation or *weak* forms. They will be derived in the next section with error estimates to show global accuracy depending on the interpolation orders. These error estimates are also used for adaptive algorithms.

Computational Formulations

The main constituents of a finite element method for the solution of electrostatics or magnetostatics are the variational or weak statement of the problem, and the approximation solution of the variational equations through the use of finite element functions. To define the weak, or variational, counterpart, we need to characterize two classes of finite element functions. The first is to be composed of trial solutions $\mathcal{S}$. The second collection of functions is called the weighting functions, or variations $\mathcal{U}$. They both have square-integrable derivatives. They are H^1-functions. The trial solutions must adhere to the Dirichlet boundary condition on Γ_d, but the weighting functions can satisfy the homogeneous counterpart of the Dirichlet boundary condition.

We now proceed with the definitions of weak formulations for both electrostatics and magnetostatics.

1. Electrostatic Solver with Error Estimate for Adaptivity

The formal statement of weak formulation of electrostatics, the counterpart of the strong form given in the previous section, goes as follows:

Given $\rho : \Omega \to \square$, $V_0 : \Gamma_d \to \square$, $\rho_{s0} : \Gamma_n \to \square$, find $\Phi \in \mathcal{S}$ such that $\forall W \in \mathcal{U}$

$$\int_\Omega \nabla W \cdot \varepsilon \nabla \Phi \, d\Omega = \int_\Omega W \rho \, d\Omega + \int_{\Gamma_n} W \rho_{s0} \, d\Gamma \tag{22}$$

The above weak form can be derived from the minimization of the electric energy [7, 8]. The crux of the finite element method is to break up every integral of the weak form into sums of elemental contributions. The method constructs the elemental matrices for each element or cell of different sizes and shapes separately and then assembles them into a global linear system in such a way that, at each element interface, the continuity of the potential and the tangential component of its gradient are *strongly* enforced. Additionally, the continuity of normal electric flux density, if there are no surface charges, is *weakly* enforced in the integral sense. Strong enforcement means that continuity is satisfied

identically at all points on the interface; weak enforcement means that continuity is naturally satisfied approximately over the interface from the minimization process, without needing to be imposed explicitly.

We now discretize the problem domain into element domains Ω^e where $\Omega = \bigcup_{e=0}^{n_{el}} \Omega^e$. In Beam Optics Analyzer, the 3D element domains are tetrahedral. Nodal points may exist anywhere on the domain but most frequently appear at the element vertices and inter-element boundaries and less often in the interiors. Discretizing the problem domain into n_{el} elements allows us to break up the integrals of Eq. (22) as follows

$$\sum_{e=1}^{n_{el}} \int_{\Omega^e} \nabla W \cdot \varepsilon \nabla \Phi \; d\Omega = \sum_{e=1}^{n_{el}} \int_{\Omega^e} W \rho \; d\Omega + \int_{\Gamma_n^e} W \rho_{s0} \; d\Gamma \tag{23}$$

Solving for Φ in Eq. (23) requires more specialization of the trial solutions and weighting functions particularly in the element domains. A typical member of the weighting functions in each element is assumed to have the form

$$W(\mathbf{x}) = \sum_{a=1}^{n_{en}} N_a(\mathbf{x}) w_a \tag{24}$$

where N_a is the interpolation or shape function associated with element node number a, w_a is a constant and n_{en} is the number of element nodes. We require throughout that they are linearly independent. Likewise for trial solutions

$$\Phi(\mathbf{x}) = \sum_{a=1}^{n_{en}} N_a(\mathbf{x}) \phi_a \tag{25}$$

where ϕ_a is the unknown at element node a (i.e., voltage) and

$$V_0(\mathbf{x}) = \sum_{a=1}^{n_{en}} N_a(\mathbf{x}) v_a \tag{26}$$

Similarly,

$$\rho(\mathbf{x}) = \sum_{a=1}^{n_{en}} N_a(\mathbf{x}) \rho_a \tag{27}$$

$$\rho_{s0}(\mathbf{x}) = \sum_{a=1}^{n_{en}} N_a(\mathbf{x}) \rho_{sa} \tag{28}$$

Substituting Eqs. (24) to (28) into Eq. (23) yields

$$\sum_{e=1}^{n_{el}}\left(\int_{\Omega^e}\nabla N_a\cdot\varepsilon\nabla N_b\ d\Omega\right)\phi_b=\sum_{e=1}^{n_{el}}\int_{\Omega^e}N_aN_c\rho_c\ d\Omega+\int_{\Gamma_n^e}N_aN_c\rho_{sc}\ d\Gamma \tag{29}$$

where repeated indices indicate implicit sums and $1\le a,b,c\le n_{en}$. Equation (29) is the finite element matrix formulation that BOA will solve for the nodal scalar potential ϕ_b.

We wish to select shape functions in such a way that, as the finite element mesh is refined, the approximate variational solution converges to the exact solution. Sufficient conditions on the shape functions for convergence turn out to be smooth on each element interior, continuous across each element boundary and complete. The first two conditions guarantee that first derivatives of the shape functions have, at worst, finite jumps across the element interfaces. This ensures that all integrals necessary for the computation of element arrays in Eq. (29) are well defined, since at most first derivatives appear in the integrals. The completeness requirement requires a shape function that is capable of exactly representing an arbitrary linear polynomial when nodal degrees of freedom are assigned to the element nodes. This will ensure the constant values over each element domain are representable as the exact solution and its first derivatives are recalculated as the finite element mesh is further and further refined.. The most common type of shape functions for electrostatics is the Lagrange polynomial. They are implemented in BOA up to fifth order, but only fully tested to quadratic interpolation.

To automatically satisfy the three basic convergence conditions, isoparametric elements are implemented for the electrostatic solver. With isoparametric elements, both spatial and field element interpolations are the same. In addition, the standard finite element formulation intrinsically ensures the continuity, although weakly (by integration instead of point wise), of the displacement vector across the element interfaces. For the piecewise linear finite element space, the discrete potential is linear, and the discrete electric field is uniform on the element interior. A procedure to smooth out the computed electric field to produce a C0-continuity finite element space will be described. We are most interested in the accuracy and smoothness of the gradient of the potential in calculating the Lorentz force in the equation of motion.

The finite element method is endowed with error estimates [7, 9, 10] that satisfy

$$\|e\|_0\le ch^{k+1}\|\Phi\|_{k+1} \tag{30}$$

where c is a constant, independent of Φ and h; k is the degree of complete polynomial appearing in the element interpolation functions; and h is the mesh parameter, a scalar characterizing the refinement of the finite element mesh. h may be taken as the diameter of the sphere that circumscribes the largest element of the mesh. The error in the finite

element approximation is denoted as $e = \Phi^h - \Phi$, Φ is the exact and Φ^h the numerical solution. The m[th] Sobolev norm of *e* is defined as

$$\|e\|_m^2 = \int_\Omega \left(e^2 + e_{,i}^2 + \cdots + e_{,\underbrace{ij\ldots k}_{m \text{ indices}}}^2 \right) d\Omega \tag{31}$$

Thus, from (30) for *linear* elements, the convergence rate in the L_2-norm for the scalar potential is second order, the same as that of the central difference scheme in the finite difference method. For *quadratic* elements, the convergence rate is third order. In the finite element method, the convergence rate of the solution can be increased as the degree of complete polynomial of the element interpolation function increases. This leads to three finite element methods: *h*, *p* and *hp*-adaptations. In *h*-adaptation, all elements have interpolation functions of the same degree, and mesh refinement or relaxation is performed when indicated by a posteriori error estimate. In *p*-adaptation [10], the mesh is kept fixed but the order of interpolation of various elements increases or decreases, depending on the posteriori error estimate. The *hp*-adaptation, as the name implies, combines both methods. Some regions of the problem domain have their elements refined or relaxed, some others have their element interpolation function order increased or decreased. In BOA, only the *h*-method is implemented, but its programming structure is designed to also work with both *p* and *hp*-adaptation.

The finite element method produces the optimal approximation from the finite element spaces. However, it is frequently the case that one is more interested in the gradient of the finite element approximation than in the approximation itself. In particle-in-cell analysis, the electric and magnetic flux density are the primary concern, rather than the potentials. As mentioned above, the discrete electric field is discontinuous across the element boundaries, meaning that the approximation of the main quantity of interest is discontinuous. For this reason, we have incorporated a post-processing procedure whereby the discontinuous approximation to the gradient of the potential is smoothed. It is important to note that even though the electric field is discontinuous across the element interfaces in the finite element formulation, the normal component of displacement vector field is weakly, in the integral sense, continuous or balanced by a surface charge. The reasons to perform such post-processing to smooth out the electric field are not purely cosmetic. One primary reason is to have the same order of interpolation for both electric field and charge density to avoid possible self force as well as increase the accuracy in computing the Lorentz force. The other main reason for post-processing of the electric field is to estimate *a posteriori* the error for mesh adaptivity. A rather natural approach to the error estimation is based on measuring the difference between the direct and post-processed (recovered) approximation to the gradient. BOA implemented this approach using the procedure developed by Zienkiewicz and Zhu [11, 12, 13, 14]. Their so-called superconvergent patch recovery (*SPR*) procedure post-processes the finite element

approximation to obtain values of the electric field at the nodes. These are the *recovered*, averaged and smoothed gradients sampled from the centroids of all elements sharing a common node.

These values are then used to obtain the globally reconstructed electric field producing a C^0-continuous gradient field. The recovered gradient is then compared with the unprocessed field gradient to obtain the posteriori error estimate.

In more detail, the *SPR* procedure creates at each vertex of the mesh a patch consisting of the elements having the same vertex. The values of the gradients of the approximation sampled at the centroids (for linear tetrahedral elements) in the patch are used to produce a recovered value at the central node by a discrete least squares fit of the values at the sampling points. The reason for sampling the gradient at the centroids of linear tetrahedral elements is because of a well known fact [16] that the gradient at the centroids is *superconvergent*. Superconvergence means that the gradient evaluated at the centroids is as good as the true value in terms of a semi-norm.

Using the recovered electric field in the place of the gradient of the *true* solution, an energy error norm can be estimated. Let $\mathbf{G}^h\left[\phi^h\right]$ denote an approximation to the gradient of the true solution obtained by the *SPR* procedure. The error estimator is then simply taken as

$$\eta^2 = \|e\|^2 = \frac{1}{2}\int_\Omega \left|\mathbf{G}^h\left[\phi^h\right] - \nabla\phi^h\right|^2 d\Omega \tag{32}$$

Having this error estimate, BOA then refines or coarsens the mesh with the goal of getting η to be smaller than a prescribed value.

Another technical issue concerns the effect of the numerical quadrature used to integrate the element integrals. In the error estimate given by Eq. (30), all integrals in the weak form must be calculated exactly. Because numerically integrated element integrals represent the rule rather than the exception, it is important to investigate the accuracy of approximate numerical integration. Strang and Fix [9] provided *sufficient* conditions for quadrature rules to maintain the full rate of convergence of the exactly integrated formulation. For linear triangles and tetrahedra, the quadrature rule must be capable of integrating polynomials of second degree, and, for quadratic triangles and tetrahedra, the rule must be capable of integrating polynomials of fourth degree. BOA follows proper quadrature rules so that the full rate of convergence of the exactly integrated formulation can be realized.

2. Magnetostatic Solver with Error Estimate for Adaptivity

The formal statement of weak formulation of magnetostatics, the counterpart of the strong form given in the previous section, goes as follows [36]:

Given $\mathbf{J}:\Omega\to\mathbb{R}^3$, $\mathbf{A}_{0t}:\Gamma_d\to\mathbb{R}^3$, $\mathbf{J}_{s0}:\Gamma_n\to\mathbb{R}^3$, find $\mathbf{A}\in\mathcal{S}$ such that $\forall\mathbf{W}\in\mathcal{U}$

$$\int_\Omega \nabla\times\mathbf{W}\cdot\frac{1}{\mu}\nabla\times\mathbf{A}\;d\Omega=\int_\Omega\mathbf{W}\cdot\mathbf{J}\;d\Omega-\int_{\Gamma_n}\mathbf{W}\cdot\mathbf{J}_{s0}\;d\Gamma \tag{33}$$

The above weak formulation can be derived from the minimization of the magnetic field energy [8, 24, 25] without explicit gauging. As indicated by Eq. (14), the source cannot be arbitrarily prescribed, but it must be divergence-free. By enforcing this compatibility condition properly, solution to the weak form (33) will converge even though its LHS matrix resulting from finite element discretization is singular having a nontrivial nullspace. The solution exists for two reasons. By requiring divergence-free sources, the RHS vector is in the range of the LHS matrix. In addition, it has been shown in [29] that the divergence of A is *weakly* defined for each iteration of the conjugate gradient method, thus it is implicitly gauged.

Thus, the prescribed current density in Eq. (33) must be divergence-free. For general geometries, prescribing such a divergence-free source term is not always possible. In order to enforce $\nabla\cdot\mathbf{J}=0$, we can search for a vector potential $\mathbf{T}$ [29] such that

$$\nabla\times\mathbf{T}=\mathbf{J} \tag{34}$$

Substituting (34) into the first integral of the RHS of (33) to obtain

$$\begin{aligned}\int_\Omega\mathbf{W}\cdot\mathbf{J}\;d\Omega&=\int_\Omega\mathbf{W}\cdot\nabla\times\mathbf{T}\;d\Omega\\&=\int_\Omega\nabla\times\mathbf{W}\cdot\mathbf{T}-\nabla\cdot\left(\mathbf{W}\times\mathbf{T}\right)d\Omega && \text{via vector identity}\\&=\int_\Omega\nabla\times\mathbf{W}\cdot\mathbf{T}\;d\Omega-\int_\Gamma\mathbf{W}\times\mathbf{T}\cdot\mathbf{n}\;d\Gamma && \text{via divergence theorem}\\&=\int_\Omega\nabla\times\mathbf{W}\cdot\mathbf{T}\;d\Omega-\int_{\Gamma_n}\mathbf{W}\times\mathbf{T}\cdot\mathbf{n}\;d\Gamma && \text{since } \mathbf{W}=\mathbf{0} \text{ on } \Gamma_d\\&=\int_\Omega\nabla\times\mathbf{W}\cdot\mathbf{T}\;d\Omega+\int_{\Gamma_n}\mathbf{W}\cdot\mathbf{n}\times\mathbf{T}\;d\Gamma && \text{via vector identity}\end{aligned} \tag{35}$$

To find $\mathbf{T}$, take dot product both sides of (34) with $\nabla\times\mathbf{W}$ and integrate to obtain

$$\int_\Omega\nabla\times\mathbf{W}\cdot\nabla\times\mathbf{T}\;d\Omega=\int_\Omega\nabla\times\mathbf{W}\cdot\mathbf{J}\;d\Omega \tag{36}$$

Thus, we will need to solve a sub-problem for the vector potential $\mathbf{T}$ given the prescribed current density $\mathbf{J}$ to ensure it to be divergence-free. Moreover, assuming $\mathbf{n}\times\mathbf{T}=\mathbf{0}$ on Γ_n and with (35) and (36), the weak formulation (33) becomes

Given $\mathbf{J}:\Omega\to\mathbb{R}^3$, $\mathbf{A}_{0t}:\Gamma_d\to\mathbb{R}^3$, $\mathbf{J}_{s0}:\Gamma_n\to\mathbb{R}^3$, $\forall\mathbf{W}\in\mathcal{U}$ first find $\mathbf{T}\in\mathcal{Z}$ to satisfy (36) then find $\mathbf{A}\in\mathcal{Z}$ such that

$$\int_\Omega \nabla\times\mathbf{W}\cdot\frac{1}{\mu}\nabla\times\mathbf{A}\;d\Omega=\int_\Omega\nabla\times\mathbf{W}\cdot\mathbf{T}\;d\Omega-\int_{\Gamma_n}\mathbf{W}\cdot\mathbf{J}_{s0}\;d\Gamma \tag{37}$$

It is noted that the weighting functions are vectors in magnetostatic unlike scalars in electrostatic finite element formulation. Similarly to electrostatics, the integrals in (37) are broken up into sums of integrals over element domains as follows

$$\sum_{e=1}^{n_{el}}\int_{\Omega^e}\nabla\times\mathbf{W}\cdot\frac{1}{\mu}\nabla\times\mathbf{A}\;d\Omega=\sum_{e=1}^{n_{el}}\int_{\Omega^e}\nabla\times\mathbf{W}\cdot\mathbf{T}\;d\Omega-\int_{\Gamma_n^e}\mathbf{W}\cdot\mathbf{J}_{s0}\;d\Gamma \tag{38}$$

To properly satisfy the continuity conditions across the element interfaces (15) , (16) and boundary conditions (12) and (13), the interpolations in the trial and weighting functions must be vector functions [24, 25, 30, 32]. Each variable in (38) can be expressed in term of the vector interpolation functions as

$$\mathbf{W}(\mathbf{x})=\sum_{a=1}^{n_{en}}\mathbf{N}_a(\mathbf{x})\,w_a \tag{39}$$

$$\mathbf{A}(\mathbf{x})=\sum_{a=1}^{n_{en}}\mathbf{N}_a(\mathbf{x})A_a \tag{40}$$

$$\mathbf{T}(\mathbf{x})=\sum_{a=1}^{n_{en}}\mathbf{N}_a(\mathbf{x})T_a \tag{41}$$

$$\mathbf{J}_{s0}=\sum_{a=1}^{n_{en}}N_a(\mathbf{x})\mathbf{J}_{sa} \tag{42}$$

It is noted that the interpolations of $\mathbf{J}_{s0}$ are scalar shape functions since its nodal values are prescribed as vectors. Substituting (39) to (42) into Eq. (38) yields the vector finite element formulation of magnetostatics

$$\begin{aligned}\sum_{e=1}^{n_{el}}\left(\int_{\Omega^e}\nabla\times\mathbf{N}_a\cdot\frac{1}{\mu}\nabla\times\mathbf{N}_b\;d\Omega\right)A_b&=\sum_{e=1}^{n_{el}}\int_{\Omega^e}\nabla\times\mathbf{N}_a\cdot\mathbf{N}_cT_c\;d\Omega\\&\quad-\sum_{e=1}^{n_{el}}\int_{\Gamma_n^e}\mathbf{N}_a.\mathbf{J}_{sc}N_c\;d\Gamma\end{aligned} \tag{43}$$

The above formulation is for linear magnetostatics in which the permeability is constant. When it varies as a function of the magnetic field as

$$\mathbf{B} = \mu(H)\mathbf{H} \tag{44}$$

Newton's method [33, 34] must be used to iteratively solve for the vector potential for nonlinear magnetostatics. To simplify derivation and implementation for nonlinear magnetostatics, we would like to work with reluctivity instead of permeability i.e.,

$$\upsilon = \frac{1}{\mu} \tag{45}$$

Thus,

$$\mathbf{H} = \upsilon(B)\mathbf{B} \tag{46}$$

where B is the magnitude of the magnetic flux density.

We begin by defining the residual by rewriting Eq. (43) after substitution of (45) as

$$\begin{aligned}\mathbf{F} \equiv & \sum_{e=1}^{n_{el}} \left(\int_{\Omega^e} \nabla \times \mathbf{N}_a \cdot \upsilon \nabla \times \mathbf{N}_b \; d\Omega \right) A_b \\ & - \sum_{e=1}^{n_{el}} \int_{\Omega^e} \nabla \times \mathbf{N}_a \cdot \mathbf{N}_c T_c \; d\Omega + \int_{\Gamma_n^e} \mathbf{N}_a . \mathbf{J}_{sc} N_c \; d\Gamma\end{aligned} \tag{47}$$

The Newton sequence is [33]

$$\mathbf{A}^{n+1} = \mathbf{A}^n - \mathbf{F}'\left(\mathbf{A}^n\right)^{-1} \mathbf{F}\left(\mathbf{A}^n\right) \tag{48}$$

where we define the Jacobian matrix by

$$\mathbf{F}'(\mathbf{A}) = \left\{ F'(\mathbf{A})_{ij} \right\} \tag{49}$$

with its element

$$F'(\mathbf{A})_{ij} = \frac{\partial F_i}{\partial A_j}(\mathbf{A}) \tag{50}$$

The computation of a Newton iteration requires

a. Evaluation of the residual $\mathbf{F}\left(\mathbf{A}^n\right)$ and test for convergence

b. Approximation solution of the equation

$$\mathbf{F}'\left(\mathbf{A}^n\right)\mathbf{S} = -\mathbf{F}\left(\mathbf{A}^n\right) \tag{51}$$

for the Newton step **S**, which is defined as

$$\mathbf{S} = \mathbf{A}^{n+1} - \mathbf{A}^n \tag{52}$$

c. Construction of $\mathbf{A}^{n+1} = \mathbf{A}^n + \lambda\mathbf{S}$, where the step length λ is selected to guarantee decrease in the Euclidian norm $\|\mathbf{F}\|$.

To complete the nonlinear magnetostatic formulation, the Jacobian matrix can be derived by taking the partial derivatives of Eq. (47).

$$\begin{aligned}\mathbf{F}'(\mathbf{A}) = &\sum_{e=1}^{n_{el}} \int_{\Omega^e} \nabla\times\mathbf{N}_a \cdot \upsilon(B) \nabla\times\mathbf{N}_b \, d\Omega \\ &+ \sum_{e=1}^{n_{el}} \int_{\Omega^e} \nabla\times\mathbf{N}_a \cdot \nabla\times\mathbf{A} \frac{1}{B}\frac{d\upsilon(B)}{dB} \nabla\times\mathbf{N}_b \cdot \nabla\times\mathbf{A} \, d\Omega\end{aligned} \tag{53}$$

BOA implements the Newton's method using the Jacobian matrix given by (53) for nonlinear magnetostatics.

The vector finite element method for magnetostatics is also endowed with the same error estimates as electrostatics, which is rewritten in term of the vector potential as

$$\|\mathbf{e}\|_0 \le ch^{k+1} \|\mathbf{A}\|_{k+1} \tag{54}$$

where $\mathbf{e} = \mathbf{A}^h - \mathbf{A}$, **A** is the exact and $\mathbf{A}^h$ the numerical solution. The same *SPR* procedure is used to post-process and smooth the magnetic flux density for computing the Lorentz force and adaptivity. The error estimate given by *SPR* for adaptivity can be rewritten for magnetostatics as

$$\eta^2 = \|\mathbf{e}\|^2 = \frac{1}{2}\int_\Omega \left|\mathbf{G}^h\left[\mathbf{A}^h\right] - \nabla\times\mathbf{A}^h\right|^2 d\Omega \tag{55}$$

where $\mathbf{G}^h\left[\mathbf{A}^h\right]$ denotes an approximation to the curl of the true vector potential obtained by the *SPR* procedure.

3. Particle Pusher

Besides solving plasma problems, where background ions and electrons are initially loaded into the problem domain with care, BOA can also solve static electric and magnetic fields to determine trajectories of charge particles through the problem domain without the background ions. In the particle trajectory analyses, BOA still solves the same Newton-Lorentz equations of motion to push particles, but only the steady state

solution is of interest. Therefore, each particle is totally independent from other particles. The time steps for each particle depend only on the mesh size to ensure that the particle does not skip a cell in its path. The static charge density field is obtained by the same scheme as above but with all particles being pushed to the end of the problem domain, accumulating charge throughout the mesh for a later update of Poisson's equation. This source field is then used in Poisson's equation to solve for the electric field. With this electric field and the prescribed magnetic field, the particle pusher again solves for all particle positions and velocities. The solution converges when both the differences in the emission current and the 1-norm of the scalar potential of the present to the previous iteration meet user-defined criteria.

With fewer restrictions on the step size in BOA, the 5^{th}-order Runge-Kutta method with the embedded 4^{th}-order formula is used to integrate the equations of motion. A high-accuracy integrator is desirable in this case so that the step size can be as large as the spatial mesh size allows. This ODE integrator allows adaptive stepsize control, which is based on the estimate of the local truncation error of the 5^{th}-order and embedded 4^{th}-order formulas [6]. Note that the Runge-Kutta method requires the immediate evaluations of the fields, and therefore are not applicable for time-dependent problems.

There are several significant differences between the trajectory and PIC analyses in BOA. Most significantly, the trajectory part of BOA solves for static fields, so the electric field solver only solves Poisson's equation. Similarly, the magnetic solver only solves for the static magnetic field produced by permanent magnets or solenoids and the self magnetic field of the particle beam. The PIC part of BOA also repeatedly solves Poisson's equation but for many temporal charge density fields (electrostatic approximation) [1]. Furthermore, in the latter case, all particles must be pushed synchronously in time.

To push particles synchronously in time, it is necessary to determine a global time step for all particles. This involves an efficient search for the smallest element on the paths of all particles. This provides the maximum distance, h_{max} that all particles are allowed to travel in that time step. The synchronized time step is then determined by $\Delta t = \min\left\{\frac{h_{\max}(t)}{v_{\max}(t)}, \frac{0.2}{\omega_p(t)}\right\}$. Since this time step is typically small and several steps may be required for a particle to pass through an element, a highly accurate ODE integrator with adaptive stepsize is not needed. Furthermore, higher order methods require either storage of additional temporal phase space data or evaluation of **E** and **B** at times other than integer or half-integer multiples of the time step with the same order of accuracy, leading to significant storage or computational expense. A more efficient integrator requiring fewer operations, such as the leapfrog method, is more desirable.

The leapfrog method applied to the equations of motion (19) and (20) can be expressed as follows [2, 3]:

$$\mathbf{u}^{t+\Delta t/2} = \mathbf{u}^{t-\Delta t/2} + \eta \Delta t \left(\mathbf{E}^t + \frac{\mathbf{u}^{t+\Delta t/2} + \mathbf{u}^{t-\Delta t/2}}{2\gamma^t} \times \mathbf{B}^t \right) \tag{56}$$

and

$$\mathbf{x}^{t+\Delta t} = \mathbf{x}^{t-\Delta t} + \Delta t \frac{\mathbf{u}^{t+\Delta t/2}}{\gamma^{t+\Delta t/2}} \tag{57}$$

It is the method of choice in PIC analysis for integrating the equations of motion. It is accurate to second order and is a two-step formula, because data from the last two steps, $\mathbf{u}^{t-\Delta t/2}$ and $\mathbf{x}^{t-\Delta t}$ are required to determine $\mathbf{u}^{t+\Delta t/2}$ and $\mathbf{x}^{t+\Delta t}$. It is very efficient because, unlike other two-step methods, it requires only one evaluation of the Lorentz force, i.e., only the field values at $\mathbf{x}^t$. The method is conditionally stable when applied to a simple harmonic oscillator $\ddot{x}(t) = -\omega_0 x(t)$, where the time step must satisfy

$$\Delta t < \frac{2}{\omega_0} \tag{58}$$

To efficiently evaluate the Lorentz force on the right hand side of (19), Boris [17] suggested the following scheme to separate the electric and magnetic forces to update **u**. The process is explained in great detail in [3] and can be summarized by the following steps:

$$\begin{aligned} \mathbf{u}^- &= \mathbf{u}^{t-\Delta t/2} + \frac{\eta \Delta t}{2} \mathbf{E}^t \\ \mathbf{u}' &= \mathbf{u}^- + \mathbf{u}^- \times \mathbf{t}^t \\ \mathbf{u}^+ &= \mathbf{u}^- + \mathbf{u}' \times \frac{2\mathbf{t}^t}{1 + \mathbf{t}^t \cdot \mathbf{t}^t} \\ \mathbf{u}^{t+\Delta t/2} &= \mathbf{u}^+ + \frac{\eta \Delta t}{2} \mathbf{E}^t \end{aligned} \tag{59}$$

where

$$\mathbf{t}^t = \frac{\eta \mathbf{B}^t}{2\gamma^t} \Delta t = \frac{\boldsymbol{\omega}_c^t}{2\gamma^t} \Delta t \tag{60}$$

in which $\mathbf{u}^+$ is a rotation of $\mathbf{u}^-$ by an angle whose value is given by $2\tan^{-1}\left|\mathbf{t}^t\right|$.

The major attraction of the leapfrog method described above is its efficiency, requiring only one function evaluation per step and only the most recent phase space data. The disadvantage is the small time step required to maintain stability. As it turns out, however, small time steps are required anyway in PIC analysis. It is desirable for particles to use several time steps to pass through each cell for accurate allocation of space charge and sampling of the fields for the integration of the equations of motion.

Adaptivity with Particles

The adaptivity procedures described earlier are based the error estimates of the field gradients. Thus, meshes in regions with high field gradients will be refined, and with low field gradients will be coarsened. In the regions with the presence of particles, fine meshes and particularly semi-structured meshes are needed. Unfortunately, the field gradients in the particle regions tend to be low; consequently, with the field-gradient-only based adaptivity procedure would actually coarsen the mesh in these regions and produce inaccurate results. The first remedy implemented in BOA is to allow the user to select surfaces or cylindrical spaces that enclose the particle regions to generate initial meshes to his specifications. The local meshes on these specified surfaces or in these cylindrical spaces are not coarsened but can be refined during adaptivity. The second remedy is that during adaptivity, all meshes having particles are kept at least the same size, but can be refined, as initially computed based on the number of launched particles. Thus, more particles in the domain give smaller initial particle mesh size. This second remedy is in effect only when adaptivity is active. The mesher in BOA also allows users to generate semi-structured meshes, so-called boundary meshes, from specified surfaces. This capability is useful when more accurate and smooth results are required nearby some pertinent surfaces.

Particle Injection

The leapfrog method requires starting values of particle velocity and position half a time step apart. Cartwright et. al. [18] developed techniques to invert the Maxwellian velocity distributions for loading or injection of particles to achieve a second-order accurate approximation to starting values. The second-order injection method proposed for this development starts with the initial particle position and velocity determined at the same time. After applying Cartwright's scheme, this results in particle positions and velocities half a time step apart and preserves the second order accuracy of the leapfrog method. For initial loading of particles from a distribution in which $\mathbf{x}(t_0)$ and $\mathbf{v}(t_0)$ are known, a general second order accurate method for arbitrary fields is given by Eqs. (24)-(25) of [18]. For particles injected from a boundary, Cartwright et al. give a number of schemes for various fields; a general scheme that maintains second order accuracy for general space and time-dependent fields is given by Eqs. (49)-(51) of [18]. Although the computational expense of these general methods is about a factor of 5-8 times greater than that of the leapfrog scheme, when the number of particles injected per step is small

compared to the number in the system volume, this should not have a significant impact on performance. Cartwright's technique of initial push relaxes the constraints on the time step, provides reduced field fluctuations due to particle statistics, and improves the accuracy of the electric field at the emitting surface. However, modification of this technique is still required for non-orthogonal boundaries.

Particle Loading

There are several procedures to load particles in **x**, **v** at $t = 0$ in BOA. These loading schemes are designed to have smooth distributions particularly in unstructured meshes to provide quiet starts. The initial conditions are usually specified in terms of particle position and velocity densities $\eta_0(\mathbf{x})$ and $f_0(\mathbf{v})$. These densities must be inverted to obtain the position and velocity $(\mathbf{x}, \mathbf{v})$ of the particles [3]. To place particles in phase space, BOA employs the following schemes:

- Particle Position
  - Spatially uniform
  - Inverse cumulative density using
    - Random number types: uniform, random, quiet
    - Probability function types: uniform, arbitrary
- Particle Velocity – Inverse Cumulative Density
  - Random number types: uniform, random, quiet
  - Probability function types: uniform, Gaussian, arbitrary

Various combinations of the above schemes can be selected via the graphical user interface in BOA to load the particles. Ions are presently implemented as stationary, initially loaded at each initial particle position but with opposite charge. Since the problem domain in BOA is arbitrary and non-orthogonal, a masking technique is used to load the particles in a virtual box that encloses the domain and keeps only those particles that remain in the domain as demonstrated by Figure 1.

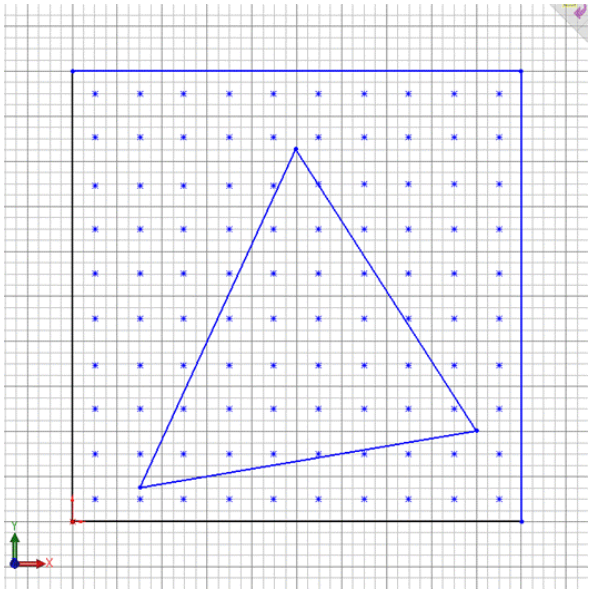

Figure 1 Loading particles onto an arbitrary geometry by masking, keeping only particles land on the domain

In the masking technique, particles are loaded using the standard schemes, but only those that land within the triangle are retained.

Thermionic Emission

The computational costs to model each electron in a given particle beam is prohibitive. It is necessary to simulate the plasma with a much smaller number of particles. Therefore, each particle is modeled as a finite-size particle with its charge modified to account for the total beam current. The effective charge of a finite-size particle is a function of the type of emitter that provides the beam current density, the number of particles emitted from that emitter, and the size of the finite elements in front of the emitter surface.

Beam current drawn from a thermionic emitter is a combination of space-charge-limited and temperature-limited electron emission. The emitter's material properties (work function distribution and applied boundary conditions), cathode temperature, electric field, and potential control the emission characteristics. Space-charge-limited current density, J_{SCL}, is calculated from the Child-Langmuir law using the electrostatic potential a distance d from the emitter surface.

$$J_{SCL} = \frac{4}{9}\varepsilon_o \sqrt{2\frac{e}{m}}\frac{\Phi^{\frac{3}{2}}}{d^2} \tag{61}$$

Temperature-limited current density is calculated from the Richardson-Dushmann equation plus the Schottky effect as follows

$$J_{SCH} = D_o T_c^2 \, \mathrm{e}^{-\frac{e}{kT_c}\left(W - \sqrt{\frac{eE_c}{4\pi\varepsilon_o}}\right)} \tag{62}$$

where $D_o = 1.2017 \times 10^6 \frac{A}{m^2 K^2}$, T_c is the absolute temperature of the cathode, e the electronic charge, k the Boltzmann's constant, W the work function, and E_c the electric field in the normal direction to the cathode surface.

Let cathode j comprise multiple patches, each with a single value of work function W_i and area a_{ji}. Also, let P_j be total the number of these patches for this emitter j. Then, combining (61) and (62), the total current emitted from the cathode j is given by Longo-Vaughan's formula

$$I_{c_j} = A_j \sum_{i=1}^{P_j} a_{ji} \frac{1}{\left[\frac{1}{J_{SCL}^n} + \frac{1}{J_{SCH}^n\left(W_i\right)}\right]^{1/n}} \tag{63}$$

where $A_j = \sum_{i=1}^{P_j} a_{ji}$ and $n = 5.5$.

Let N_j be the number of electrons emitted from the cathode j and Δt_0 be the initial time step specified for these electrons. The effective charge for each electron emitted from the thermionic cathode j is

$$q_j = \frac{I_{c_j}}{N_j} \Delta t_o \tag{64}$$

In constant-current emitters, the user specifies either a distribution of total constant current or constant current density for a cathode j. Equation (64) is used to calculate the effective charge for particles emitted from cathode j.

When thermal effects need to be accounted for, the Child-Langmuir law, Eq. (61) requires some modification via

$$J_{SCL} = \frac{4}{9} \varepsilon_o \sqrt{2\frac{e}{m}} \frac{\Phi^{\frac{3}{2}}}{d^2} G_T \tag{65}$$

where the temperature correction factor G_T derived from the modified Maxwellian energy distribution is approximate by an empirical formula

$$G_T \approx 1 + 0.02468\left(\frac{T_c}{\Phi}\right)^{1/2} - 0.00197\left(\frac{T_c}{\Phi}\right)^{3/4} \tag{66}$$

In the relativistic region, Eq. (61) requires further modification with

$$J_{SCL} = \frac{4}{9}\varepsilon_o \sqrt{2\frac{e}{m}}\frac{\Phi^{\frac{3}{2}}}{d^2} G_R \tag{67}$$

where the relativistic correction factor is approximately

$$\begin{aligned} G_R \approx 1 - 0.107(\gamma - 1) + 0.024(\gamma - 1)^2 - 0.007(\gamma - 1)^3 + 0.002307(\gamma - 1)^4 \\ - 0.0008229(\gamma - 1)^5 \end{aligned} \tag{68}$$

For thermionic emitters, specifying the particle initial velocities is more complicated when thermal velocity effects are present. To formalize the procedure for thermal effects, let $2P+1$ be the number of particles emitted at each point or site on the cathode surface, and let *M* be the number of energy levels for each particle. Thus, there will be a total of $M(2P+1)$ electrons launched at each site. It is noted that each particle for a given cathode has the same effective charge. Let the launch angle θ_i be the angle between the particle velocity and the normal unit vector, **n**, to the emitter surface. Also assume that the launch angles, θ_i's, are symmetric with respect to the normal unit vector, i.e.,

$$\theta_i = \begin{cases} 0, & i = 0 \\ \theta_{-i}, & i = 1, \ldots, P \end{cases} \tag{69}$$

By setting $P = 0$ is to neglect the thermal effect. We now proceed to determine the v_i's. It has been predicted theoretically, and verified experimentally, that the velocity distribution of electrons emitted from a thermionic emitter has a Maxwell-Boltzmann distribution. By using the concept of a phase-space-density distribution, which is a Gaussian distribution and a function of six variables - three of positions and three of velocities, the mean values of various quantities can be derived. At the cathode, all permissible electron velocities have the following mean values

$$\overline{v^2} = \frac{3}{2}\frac{2kT_c}{m}, \qquad \overline{v_t} = 0 \tag{70}$$

The second average implies the tangential components of all possible velocities at a given launch site are symmetric with respect to **n** as long as there are *p* particles on each side and their launch angles are also symmetric. There is only one average left; thus, there will be only one possible unknown. Let it be v_{00}, the normal velocity at 0^{th} energy level. Let's further set

$$v_{ij} = c_i d_j v_{00} \tag{71}$$

Substituting (71) into (70) yields

$$\sum_{j=0}^{M-1} d_j^2 v_{00}^2 + 2\left(\sum_{i=1}^{P} c_i^2\right) d_j^2 v_{00}^2 = M\left(2P+1\right)\overline{v^2} \tag{72}$$

With the properties $c_0 = d_0 = 1$, we can solve (72) for v_{00}

$$v_{00}^2 = \frac{M\left(2P+1\right)\overline{v^2}}{\left(\sum_{j=0}^{M-1} d_j^2\right)\left(2\sum_{i=0}^{P} c_i^2 - 1\right)} \tag{73}$$

where $\overline{v^2}$ is given by (70). In addition, a randomizer can select c_i's and d_j's with $1 \le i \le P,\ 1 \le j \prec M$ and $c_i, d_j \le 1$. Equations (70), (71) and (73) with the randomly selected c_i's and d_j's will provide the initial thermal velocities required to launch particles from an emitter.

Secondary Emission

When energetic electrons strike a surface of a solid, two possible scattering events occur. The first type of scattering process is elastic ensuing from Coulomb forces between the incident electron and the atoms of the solid. Large angular deflection of the incident electron can be the result, but its energy is very much unchanged. The second type of scattering process is inelastic in which the incident electron dissipates its energy to the solid but has small angular deflection.

One of its secondary emission options, BOA allows both types of scattering by letting the user specify the secondary emission yield coefficient for a prescribed *secondary emitter* surface. The prescribed yield coefficient could be obtained from other external means, such as experiments or published literature, and most likely including both scattering effects. In addition, BOA also provides an option to internally calculate the yield coefficient, but only for the inelastic backscattering. We will discuss more about the calculation of the backscattering electrons and the yield coefficient shortly.

To prevent endless scattering and reduce computational costs associated with secondary emission, a secondary emitter will also need a prescribed number of secondary electrons per incidence and number of permissible secondary generations. In BOA, the very first incident electron is considered as the 0^{th} generation; its first scattering electrons are of the 1^{st} generation; and when these 1^{st} generations impinge another secondary surface and cause more scattering electrons, they are designated the 2^{nd} generation secondaries; and so on.

In the first option, the elastic and inelastic scattering combo, for each incident electron arrives at a secondary emitter surface with its specific kinetic energy and angle. BOA selects the number of scattering electrons equal to the prescribed number of secondary electrons per incidence and assigns them randomly with velocities and deflection angles. In doing so, BOA conserves the incident energy after considering the prescribed yield coefficient. However, BOA will always set the defection angle of one of the scattering electrons to be simply a reflection of the incident angle with respect to the plane normal.

In the second option, or inelastic scattering, BOA implements a Monte Carlo plural scattering algorithm that accurately and efficiently predicts yield, trajectories and energies of backscattered electrons [43]. For each incident electron, the algorithm takes into account its energy and incident angle, the atomic number, atomic weight and density of the solid to determine a statistically reasonable path for the electron through the solid from one scattering event to the next until it leaves the solid or loses all of its energy.

The basis of the plural scattering algorithm is to first calculate the Bethe range, that is, the total distance the electron will travel through the solid, and given by

$$R_B = \int_0^{E_0} -\frac{1}{\frac{dE}{ds}}\,dE \tag{74}$$

where E_0 is the energy of the incident electron, E is instantaneous electron energy, both in keV, s is the distance along the particle trajectory, and $\frac{dE}{ds}$ is related to the Bethe stopping power as follows

$$\frac{dE}{ds} = \rho\frac{dE}{dS} \tag{75}$$

where ρ is the solid density in g/cc and $S = \rho s.$ The Bethe stopping power can be estimated by

$$\frac{dE}{dS} = -78,500\frac{Z}{AE}\ln\left(1.166\frac{E+0.85J}{J}\right) \tag{76}$$

where Z and A are the atomic number and atomic weight of the solid, and J in keV is given by

$$J = 0.001\left(9.76Z + \frac{58.5}{Z^{0.19}}\right) \tag{77}$$

The Bethe range is then divided into an arbitrary but sufficient number of equal length steps to give the distance from one scattering event to the next. The energy of the electron at n^{th} step can then be calculated from the Beth stopping power and the previous step as follows

$$E_n = E_{n-1} - \int_{s_{n-1}}^{s_n} \rho \frac{dE}{dS}\, ds \tag{78}$$

The scattering angle is given by a randomized Rutherford-type scattering formula

$$\tan\frac{\phi}{2} = \frac{F}{E}\left(\frac{1}{\sqrt{R_p}} - 1\right) \tag{79}$$

where $0 \le R_p \le 1$ is a random number,

$$F = 0.0072\frac{ZE_0}{P_0} \tag{80}$$

and

$$P_0 = 0.394Z^{0.4} \tag{81}$$

Finally the azimuthal scattering angle with equal probability of scattering can be calculated from

$$\psi = 2\pi R_a \tag{82}$$

where $0 \le R_a \le 1$ is another random number.

Given the electron coordinates, step length, scattering and azimuthal angles at the $(n-1)^{th}$ step, these can be updated to the n^{th} step. The process continues until either the number of specified maximum steps is met or when the electron leaves the solid.

When BOA samples a large number of incident electrons per secondary emitter, the above algorithm provides statistically accurate results, in excellent agreement with the measured backscatter coefficients, which is the ratio of the number of backscattered to incident electrons, as well as measured angular and energy distributions [43].

Particle Boundaries

In plasma simulations, it is very desirable to model boundaries that continually emit or absorb particles to reduce the problem size and computer resources. Particularly with

geometries exhibiting symmetries, particle reflecting planes would be highly advantageous. BOA implements three types of particle boundaries:

a. **Absorbing Boundary:** a particle crosses an absorbing boundary and is removed from the domain,

b. **Reflective Plane:** a particle crosses a reflective plane, and its velocity is reflected back with respect to the plane normal,

c. **Periodic Planes:** a pair of planes composes a periodic boundary such that, when a particle crosses one plane, its position is translated back into the domain at the other plane.

Allocation of Charges and Current Density Vectors

In BOA, we implemented and fully tested the first order interpolation of the particle charge from its position in a tetrahedron to the vertices. In 2D, in which an unstructured mesh consists of triangles, the first order interpolation scheme can be physically interpreted as an area weighting scheme [3]. As shown in Fig. 2, a particle located at point *i* within the triangle *abc* would have its charge deposited to vertex *b* by

$$q_b = q_i \frac{A(i,c,a)}{A(a,b,c)}, \tag{83}$$

which is exactly the same as using the linear Lagrange interpolation function of vertex *b* evaluated at point *i* with the local, triangular coordinates $\boldsymbol{\xi}_{\mathbf{i}}(\mathbf{x}_i)$

$$q_b = q_i N_b(\boldsymbol{\xi}_i). \tag{84}$$

The procedure is readily generalized to 3D geometries. Instead of triangle areas, tetrahedron volumes are substituted in Eq. (83), and the linear Lagrange interpolation function for tetrahedra is used in Eq. (84).

This linear interpolation of charge is extremely convenient within the finite element framework. All the information needed is confined to each element containing the particle. No information outside of the cell in question is required. Any higher orders of charge interpolation would also be required to meet this condition, that is, only data within the cell in question is needed, to be useful by the existing implementation of the finite element method.

All charge quantities mentioned in Eq. (83) and (84) are computer charge. In the current implementation of BOA, the charge on macro-particles is simply a fixed particle weight, $w = q_c / q_p$, based on the number of particles per cell chosen to satisfy statistical

requirements. Here, q_c is the charge on the computer particle, and q_p is the charge on the physical particle.

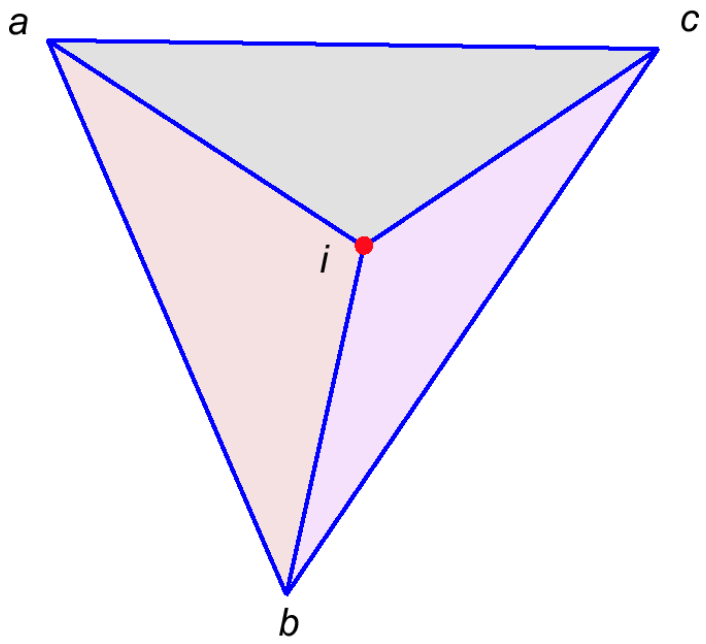

Figure 2 Allocation of charge of particle positioned at *i* to vertices *a*, *b* and *c* based on the area weighting. Charge deposited to vertex *b* is weighted by the area *ica*

After the charges of all particles are allocated and accumulated at all vertices of the mesh, the charge density at each vertex is calculated. The consistency condition for a proper charge density calculation is as follows: If point charges of a uniform charge density field are deposited and accumulated on an unstructured mesh, the numerical density field must be as close to uniform as the analytical field. In another words, the method of calculating the charge density should recover the exact distribution in the limit of the mesh size approaching infinitesimal.

The symmetric spline weighting method for charge and current density, which was developed recently by Verboncoeur [21, 35], not only meets the mentioned consistency condition but also has other desirable properties, such as conservation of charge and generality on unstructured meshes with arbitrary particle-mesh interpolations. Consider that a patch consists of the M elements having the same vertex a with the total charge q_a deposited and accumulated at the central vertex. Instead of calculating the charge density at vertex a as

$$\rho_a = \frac{q_a}{\sum_{m=1}^{M} \Omega_m} \tag{85}$$

where Ω_m is the volume of the m^{th} element sharing the same vertex a, the symmetric spline weighting scheme modifies Eq. (85) as follows

$$\rho_a = \frac{q_a}{\sum_{m=1}^{M} \int_{\Omega_m} N_a(\xi)\, d\Omega} \tag{86}$$

Here N_a is the interpolation function of node a evaluated at $\boldsymbol{\xi}$. To use Eq. (86) one would must evaluate the integrals in the denominator. It is simpler for linear triangles and tetrahedral where the integrals can be derived analytically. Then,

$$\rho_a = \frac{q_a}{\alpha \sum_{m=1}^{M} \Omega_m}, \tag{87}$$

where $\alpha = 3$ for triangles and $\alpha = 4$ for tetrahedra. Both linear and higher orders of charge interpolation are implemented in BOA.

For self magnetic field analysis, the current density is required for the magnetostatic field solver. In BOA the current is weighted by a charge conserving method using the same interpolation for charge of a particle at $\boldsymbol{\xi}_{\mathbf{i}}(\mathbf{x}_i)$ as follows

$$q_a \mathbf{v}_a = q_i \mathbf{v}_i N_a(\boldsymbol{\xi}_i) \tag{88}$$

$$\mathbf{J}_a = \frac{q_a \mathbf{v}_a}{\sum_{m=1}^{M} \int_{\Omega_m} N_a(\xi)\, d\Omega} \tag{89}$$

where q_a, $\mathbf{v}_a$ and $\mathbf{J}_a$ are, respectively, the charge, velocity and current density at node a. It is noted that q_a and $\mathbf{v}_a$ are interpolated as a single product quantity by Eq. (88) as required by the charge conserving scheme.

New Robust Ray Tracing Routine

BOA implemented the Algorithm Oriented Mesh Database (AOMD) [37, 38] to efficiently maintain the 3D mesh data allowing the retrieval of every possible set of entity adjacencies without having to do a global traversal of the graph of the mesh. The 3D octree data structure is the backbone of AOMD allowing the efficient search of a tetrahedron for a given particle position in a constant order [39, 40] to obtain the field

information for the Lorentz force. However, a particle pusher for unstructured meshes still requires ray-tracing routines to determine the next possible element the particle will travel to from its existing position and velocity. This is needed when the particle arrives near electrodes or the problem boundary and to estimate the mesh based time steps. BOA has included the standard ray tracing routines that work well in unstructured meshes but cell sizes within an order of magnitude of each other. When tetrahedra of drastic different sizes exist in the problem domain, due to round-off error, the routine requires care to handle the boundary cases. An example is when the ray is parallel to a tetrahedral face and passes through a vertex or intersects an edge.

The main functions of a ray tracing routine in an unstructured mesh are to find a triangular face of a tetrahedron that a ray will intercept and to determine the coordinates of the interception. The standard ray tracing technique [41] implemented earlier in BOA can be summarized as follows:

The ray equation in the parametric form can be expressed as

$$\mathbf{r}(t) = \mathbf{p} + t\mathbf{d} \tag{90}$$

where **p** is the ray origin and **d** is its direction. The plane equation that contains a triangle can also be expressed implicitly as

$$\mathrm{f}(\mathbf{x}) = (\mathbf{x} - \mathbf{a}) \cdot \mathbf{n} = 0 \tag{91}$$

where **x** is any point on the plane, **n** is its unit normal vector, and **a** is a fixed point on the plane. A ray intersects a plane if and only if

$$\mathrm{f}(\mathbf{r}(t)) = (\mathbf{p} + t\mathbf{d} - \mathbf{a}) \cdot \mathbf{n} = 0 \tag{92}$$

Solving Eq. (92) for *t*,

$$t = \frac{(\mathbf{a} - \mathbf{p}) \cdot \mathbf{n}}{\mathbf{d} \cdot \mathbf{n}} \tag{93}$$

The ray intercepts the plane given by Eq. (91) only if $t \geq 0.0$. Once the ray-plane interception has been determined, we still have to check if the intercepted point is inside the triangle by projecting both the point and the triangle on a 2D plane. This 2D plane is chosen based on the largest component of the unit normal. The difficulties of using Eq. (93) arise when either its numerator or denominator or both is close to zero. When only the numerator is zero, the ray origin, **p**, is on the plane. When only the denominator is zero, the ray is parallel but not intercepting the plane. When both numerator and denominator are zero, the ray lies in the plane. With floating point operations, a zero criterion must be set for testing these boundary cases. Since **a** and **p** are in global

coordinates based on the physical dimensions of the problem, and if the mesh cells vary drastically in size, a universal zero criterion is difficult to determine.

A more robust ray tracing method, based on the triangular coordinates, was given by [42] and implemented into BOA. Figure 2 briefly illustrates the triangular coordinates. The triple α, β, γ defines triangular coordinates. Each is zero along one edge and takes on value 1 as the opposite vertex. Thus,

$$0 \le \alpha,\ \beta,\ \gamma \le 1 \tag{94}$$

Moreover,

$$\alpha + \beta + \gamma = 1 \tag{95}$$

A point **x** in the triangle **abc** must satisfy the following equation

$$\mathbf{x} = \alpha\mathbf{a} + \beta\mathbf{b} + \gamma\mathbf{c} \tag{96}$$

Substituting Eq. (95) into (96) yields

$$\mathbf{x} = \mathbf{a} + \beta\left(\mathbf{b} - \mathbf{a}\right) + \gamma\left(\mathbf{c} - \mathbf{a}\right) \tag{97}$$

Then a triangle interior test for any point **x** is

$$\beta > 0; \qquad \gamma > 0; \qquad \beta + \gamma < 1 \tag{98}$$

Replacing **r** by **x** in Eq. (90) and substituting the result into (97) yield a system of three linear equations

$$\left[\mathbf{b} - \mathbf{a} \quad \mathbf{c} - \mathbf{a} \quad -\mathbf{d}\right]\begin{Bmatrix}\beta\\ \gamma\\ t\end{Bmatrix} = \mathbf{p} - \mathbf{a} \tag{99}$$

Solving the above 3×3 matrix equation gives the ray parameter *t* and the triangle coordinates of the intersection in *one* step. The LHS matrix would be singular if the ray lies in the plane that contains the triangle. Singularity test of the LHS matrix can easily be done if Eq. (99) is solved by Cramer's rule. In addition, tests for boundary cases and triangle interior are now all based on triangular, normalized coordinates permitting the setting of a universal zero condition.

Simulations with Graphical User Interface

Much emphasis was given to the development of the graphical user interface for BOA. The GUI is intuitive, user friendly, and feature rich. To demonstrate the usage and

capabilities of the GUI, we will show how it is used to import the geometry, set up the model, create the initial mesh, execute the analysis, post-process and visualize the results. We begin with the starting screen of the GUI in Fig. 3 showing the top menu. The user normally works from left to right.

In the file menu, where a project and its associated cases can be accessed, we will open an existing project, Boa Cases and create a new case, JLabSelf, as shown by Fig. 4. In the *AddCase* menu, we can specify the analysis type and add comments. The analysis type can also be changed later in the main menu, under *Analysis*. We click on the Model menu to open the *Geometry Parameters* window, shown in Fig. 5. Here we can browse to the existing locations of the CAD drawing and model files, which is in ACIS or Parasolid format and usually generated by a CAD package. The model file is copied to the case directory. The CAD drawing can also be opened with the *CAD…* button. We can check the model length unit by the *Verify* button. The model unit of this klystron electron gun is inches.

There are several ways to model the background in BOA. If the domain of interest consists of only one region (part) and boundary conditions can be specified on the region surfaces, then the *None* option should be selected. If the model consists of many parts, then, in general, the default *Infinite* option should be selected, as we have done in Fig. 5. However, if the model possesses symmetries, other background options are available depending on the types of symmetry. For example, if the model consists of several concentric spheres, then it possesses symmetry in all three coordinates. The *x,y,z Semi-Infinite* option would be the choice. If the model has multiple concentric cylinders then there are several background choices. Either a half of the geometry can be modeled with the *x Semi-Infinite* option if the axis of the cylinders is the z axis, or a quarter of the geometry can be modeled with the *x,y Semi-Infinite* background.

The next step is to assign boundary conditions and material properties for components via the *Electrostatics* window of the *Attributes* menu. Besides setting the boundary conditions and material properties, this window also allows settings of the part color, visibility, and opacity. Since this is an electrostatic beam with self magnetic field analysis, both the submenus, *Electrostatics* and *Beam Optics* are available. If this were a Magnetostatic or Electrostatics analysis, only the submenu *Magnetostatics* or *Electrostatics* would appear. In addition, the *View* menu can be used to change the geometry orientation, slice the model as seen in Fig. 6, and set the axis type, background color and variety of other parameters.

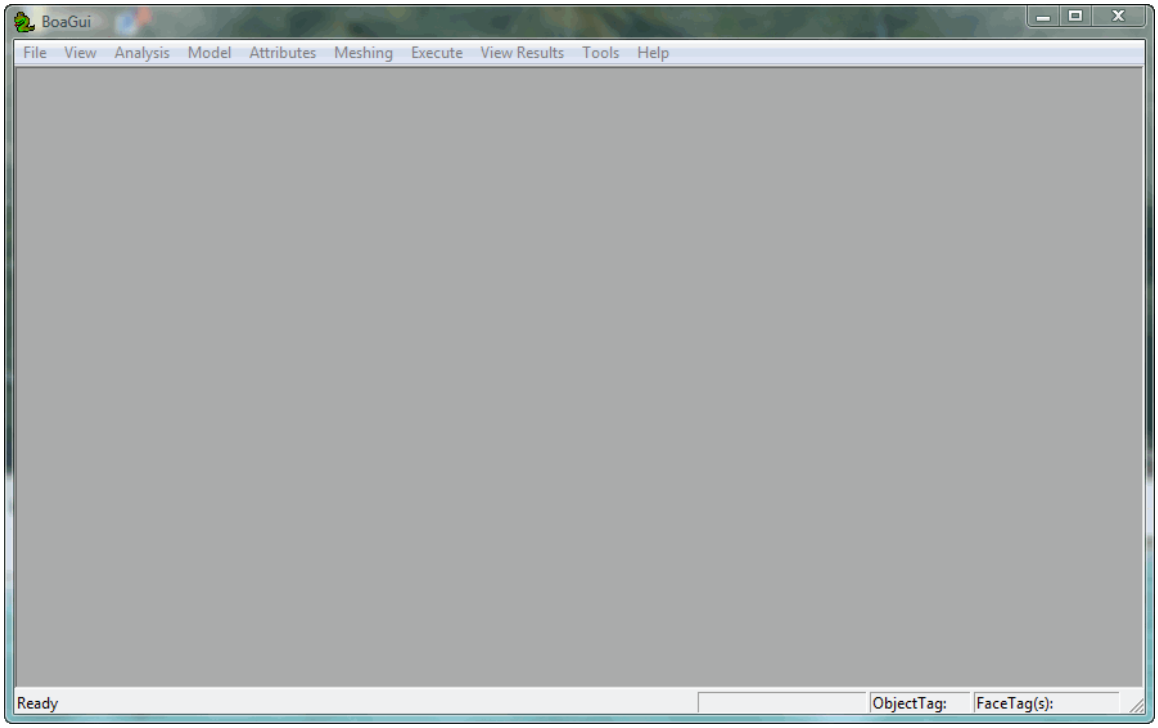

Figure 3 The opening screen of the GUI. User usually starts the simulation with the top menus from left to right

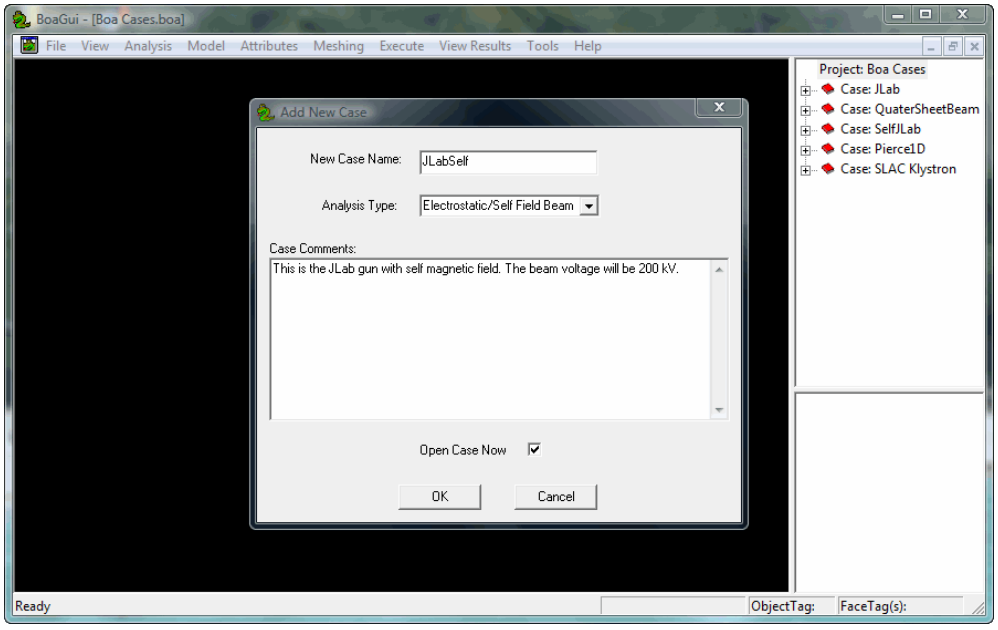

Figure 4 Use *Add New Case* menu to specify the analysis type, and enter comments

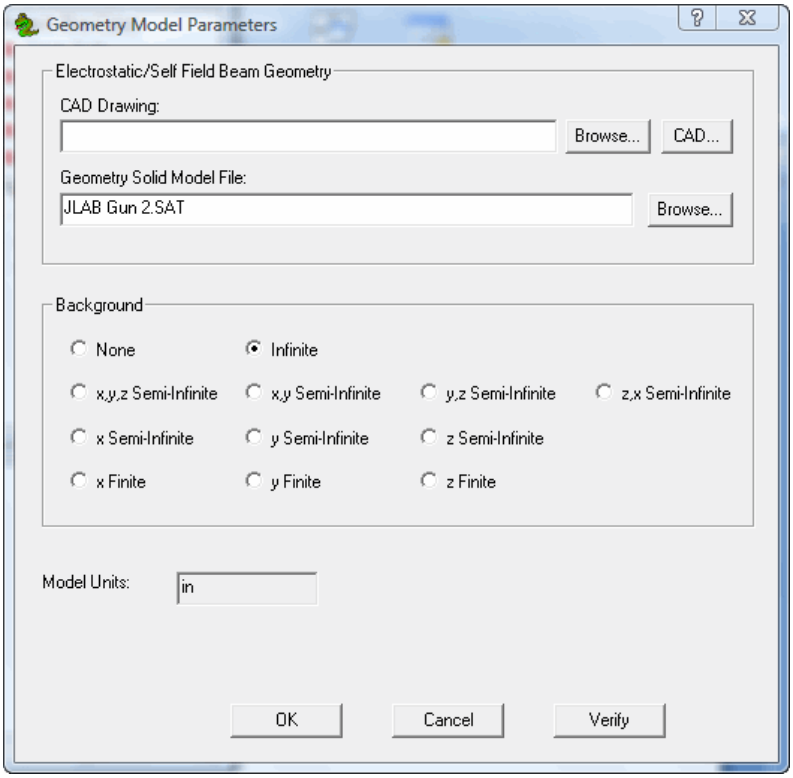

Figure 5 Use *Model* screen to specify the location of the model file, select the background type and verify the model unit

The klystron electron gun consists of the emitter, focus electrode, mod anode and the anode. Both the emitter and the focus electrode, as shown in Fig. 6, are set at -200 kV, and the other two electrodes are set to ground potential. We continue to specify the emitter properties in the Beam Optics submenu shown in Fig. 7. In this window, the user can set the emitter type, convergence criteria, the maximum number of secondary emission generations, particle loading schemes, particle boundaries, optical transparent regions, etc. A thermionic emitter on the orange colored surface is specified as shown in Fig. 8. Approximately 200 particles are requested to be launched without thermal effects. The cathode material and temperature are also prescribed here.

We now proceed to specify the parameters for the initial mesh by clicking on the *Meshing* top menu and select *Parameter…* As mentioned in the section Adaptivity with Particles, to ensure the meshes are not coarsened in the particle regions, we will locally specify mesh size on pertinent surfaces, particularly on the beam tunnel of the mod anode and anode (highlighted in yellow) as shown in Fig. 9.

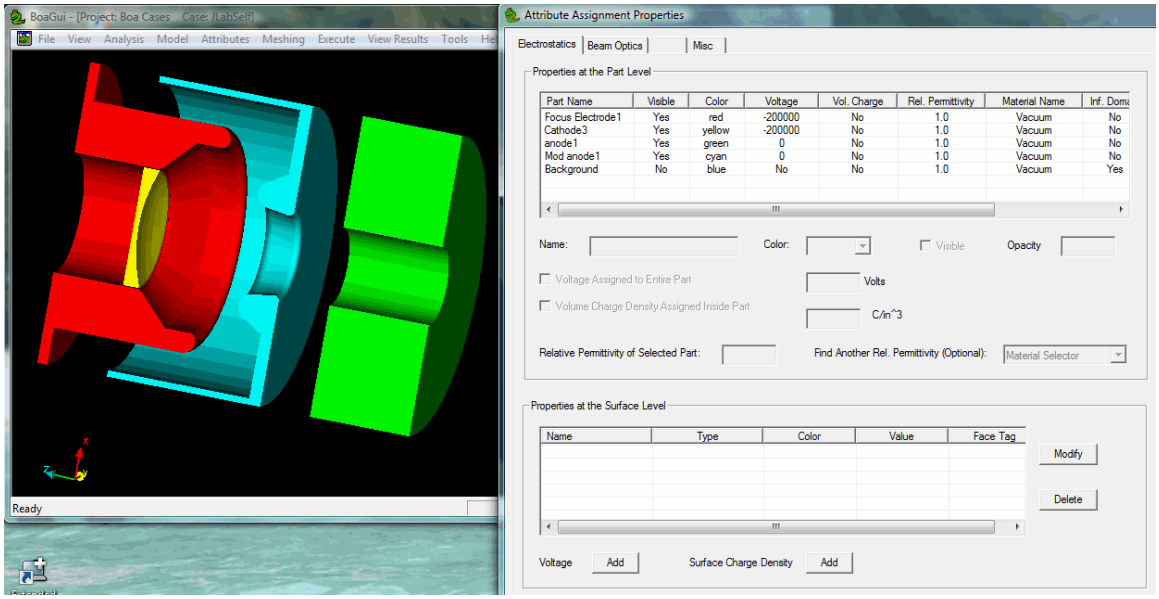

Figure 6 Use the *Electrostatics* window of the *Attributes* menu to set the boundary conditions and material properties

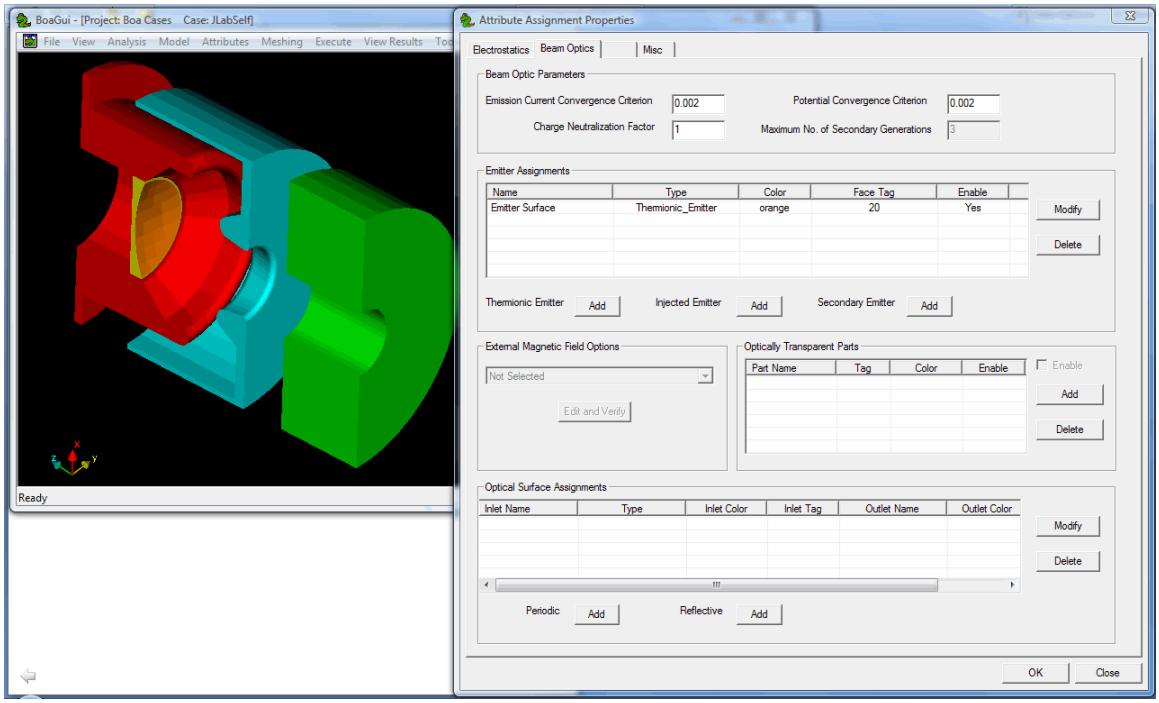

Figure 7 Use the submenu *Beam Optics* to specify the emitter properties, import external magnetic field, particle boundaries, particle loading etc.

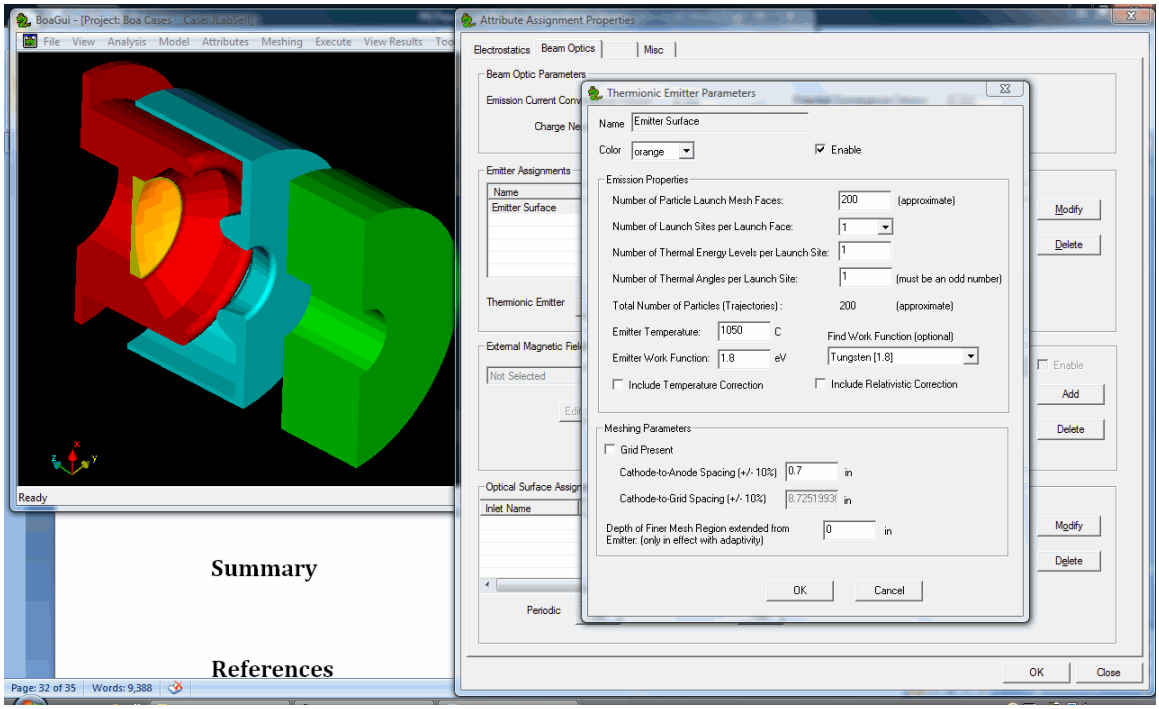

Figure 8 Use *Add* or *Modify* button to change properties of a selected emitter. The above window shows the properties of a thermionic emitter

After setting the meshing parameters and closing the window, we can generate the initial mesh by clicking again on the *Meshing* menu and selecting *Generate Initial Mesh*. During the mesh generation, a status window appears showing the progress, as shown in Fig. 10. If the user wants to view the same meshing log later, he can check on the *Create Log File* option in the *Mesh Parameters* window before generating the initial mesh. This initial mesh is sliced and displayed in Fig. 11 with its attributes: number of regions, faces, edges and vertices. The mesh includes fine mesh in the beam tunnel, but somewhat coarser mesh in between the cathode-mod anode gap. The mesh is very coarse elsewhere.

At this point we have completed all the steps to set the material properties and boundary conditions, and generate the initial mesh. We still need to specify the execution parameters, including those for adaptivity. In the *Execute* menu, shown in Fig. 12, we specify linear interpolation, number of adaptivity passes of 2, and use the default value of the adaptivity objective of 0.25. The smaller adaptivity objective results in finer mesh in the high field gradient regions.

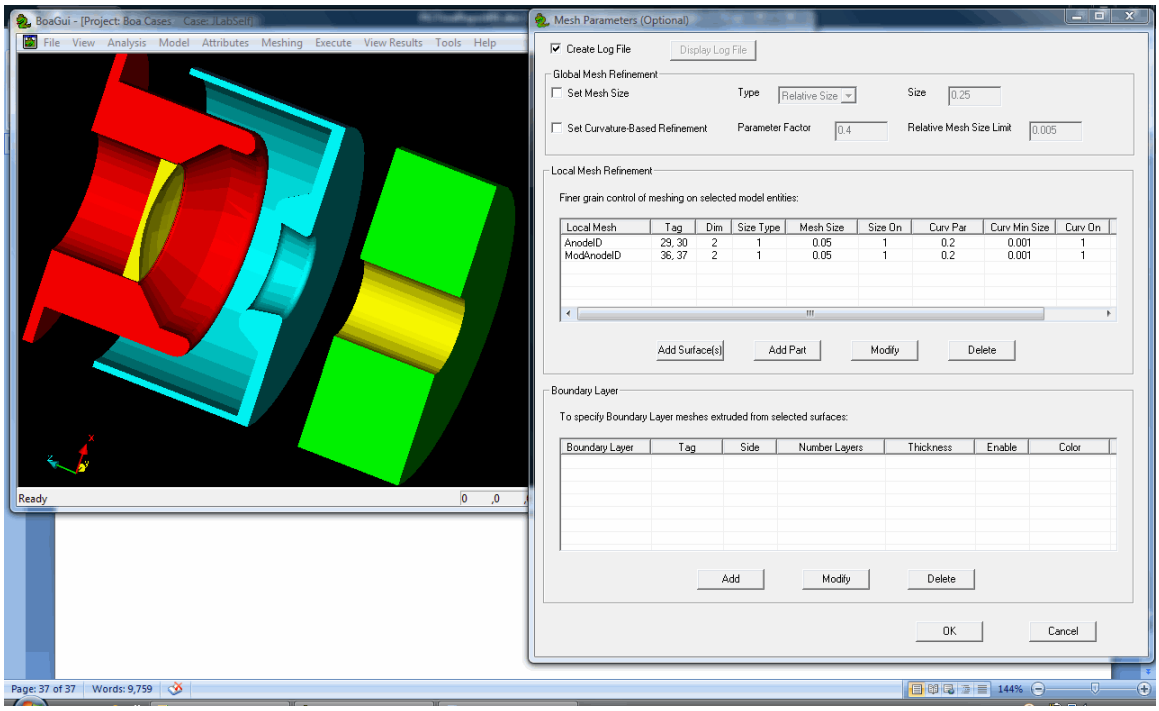

Figure 9 Use *Meshing* menu to specify mesh size and other meshing parameters

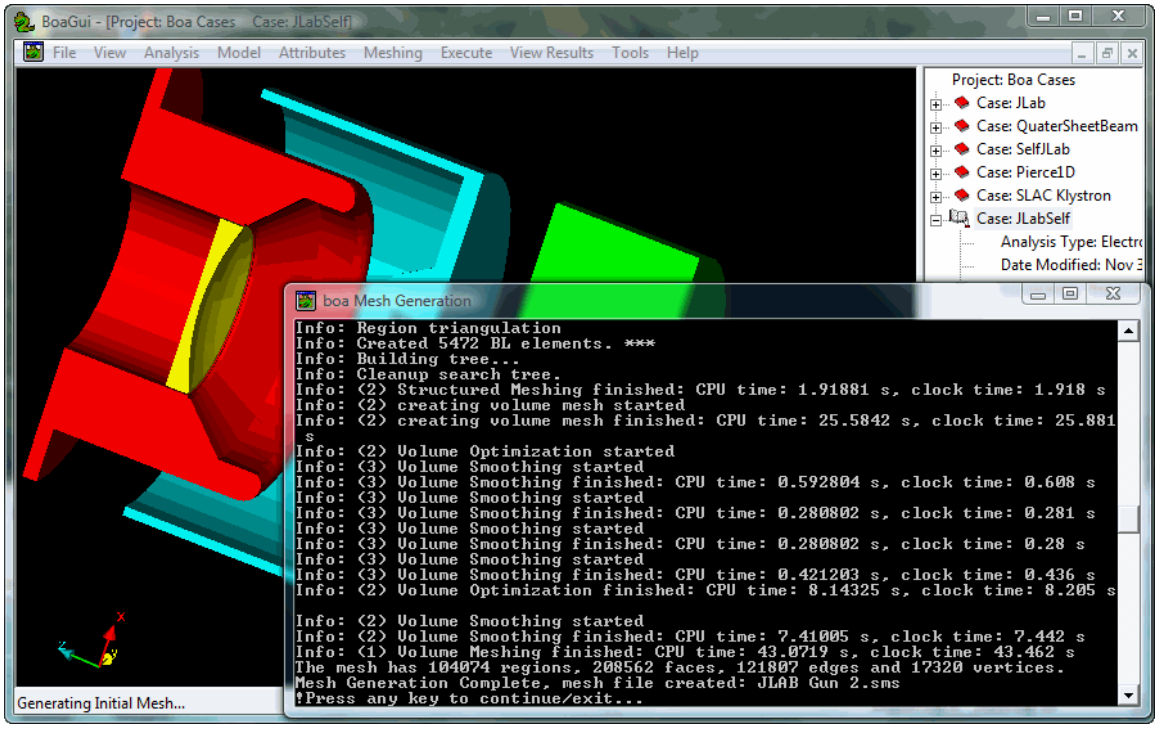

Figure 10 Status of the meshing of the initial mesh

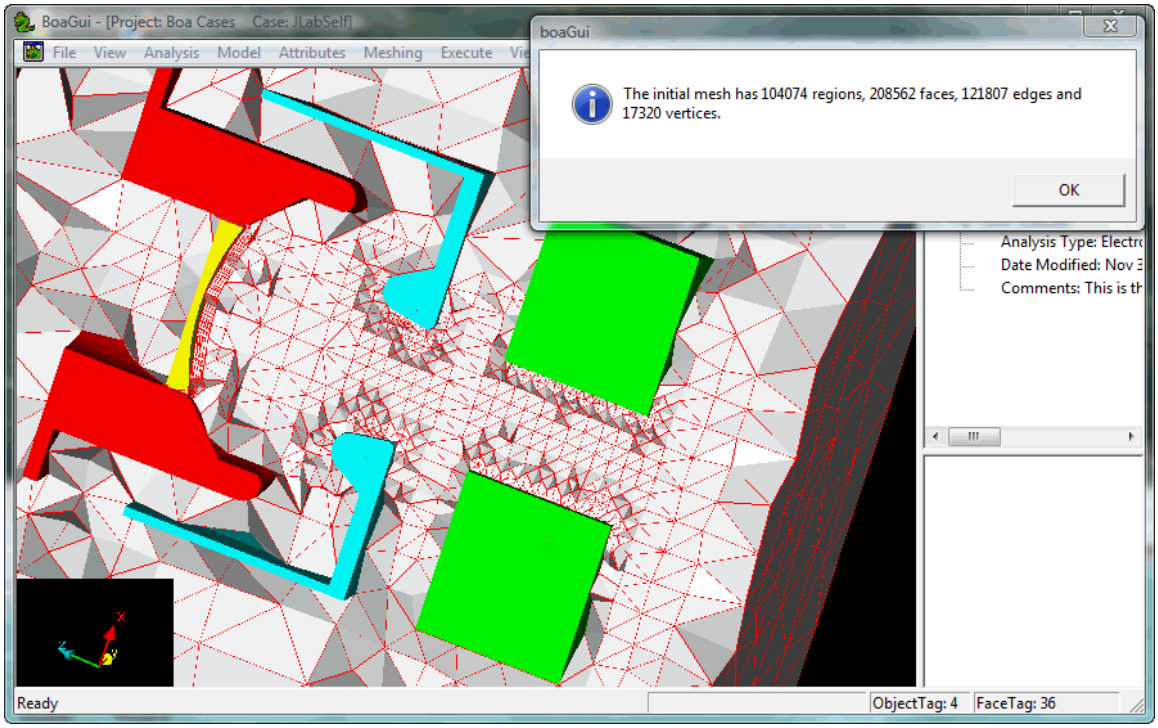

Figure 11 This initial mesh has fine mesh nearby the beam tunnel of the mod anode and anode but coarse elsewhere. It has 104074 tetrahedra and 17320 vertices

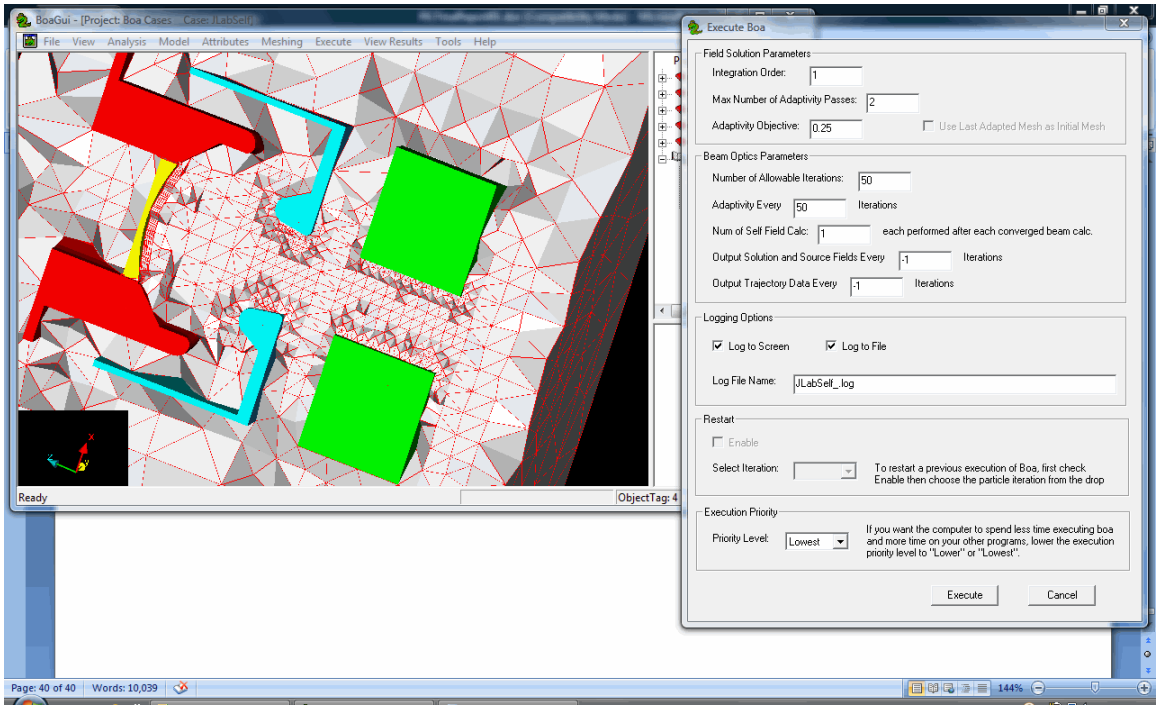

Figure 12 Use Execute menu to set interpolation order, adaptivity and other execution parameters

We also set the number of allowable particle iterations and the adaptivity iterations to the same value. Setting this way, adaptivity is only performed in the first three iterations, and

then the last adapted mesh would be kept invariant for the remainder of the analysis. For the self field analysis option, we request two updates of the self magnetic field, each after an electrostatic beam convergence. We are now ready to use the *Execute* button to begin the simulation. During the simulation, a log window shows the progress of the simulation, including a plot of the emission current and field convergence in the lower right corner of the screen, as seen in Fig. 13. Since BOA is multithreaded, we can relist the run log in a separate window, recheck the problem boundary conditions and redisplay the initial mesh while the simulation is proceeding. One can also abort the execution, and, depending on how the storing of the field and particle solution is set, one can restart the case later from the last available saved iteration.

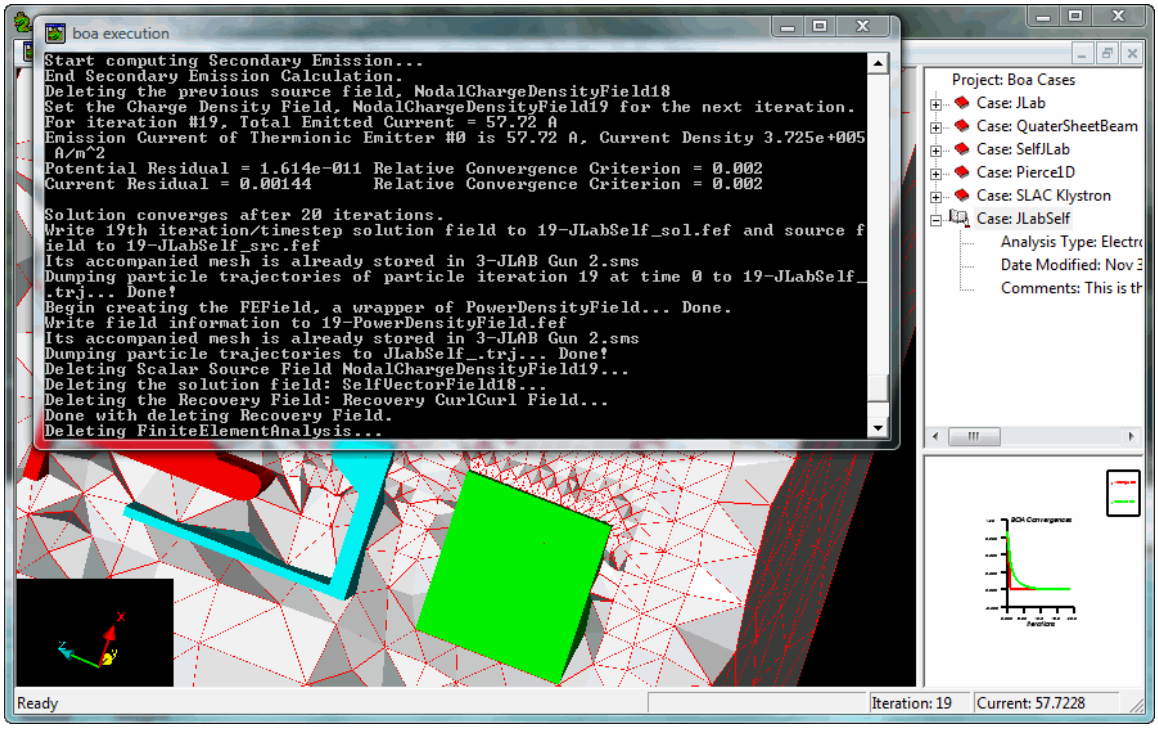

Figure 13 Status window of the simulation and convergence plot in the lower right corner

Much effort was devoted to the post-processing capability of BOA. An electrostatic beam with self magnetic field analysis is a good candidate to showcase BOA's post-processing capabilities, since all three post-processing types: electrostatics, magnetostatics and beam optics can be shown in the same case. We start with the electrostatic fields by clicking on the *View Results* menu and selecting *Electrostatics*. In the *Electrostatic Display Control* panel, shown in Fig. 14, one can process and display voltage, electric field, charge density in a region, surface or along a line in space, as long as its end points are within the problem domain. One can also slice the displayed fields for clearer visualization or for lower dimension contouring.

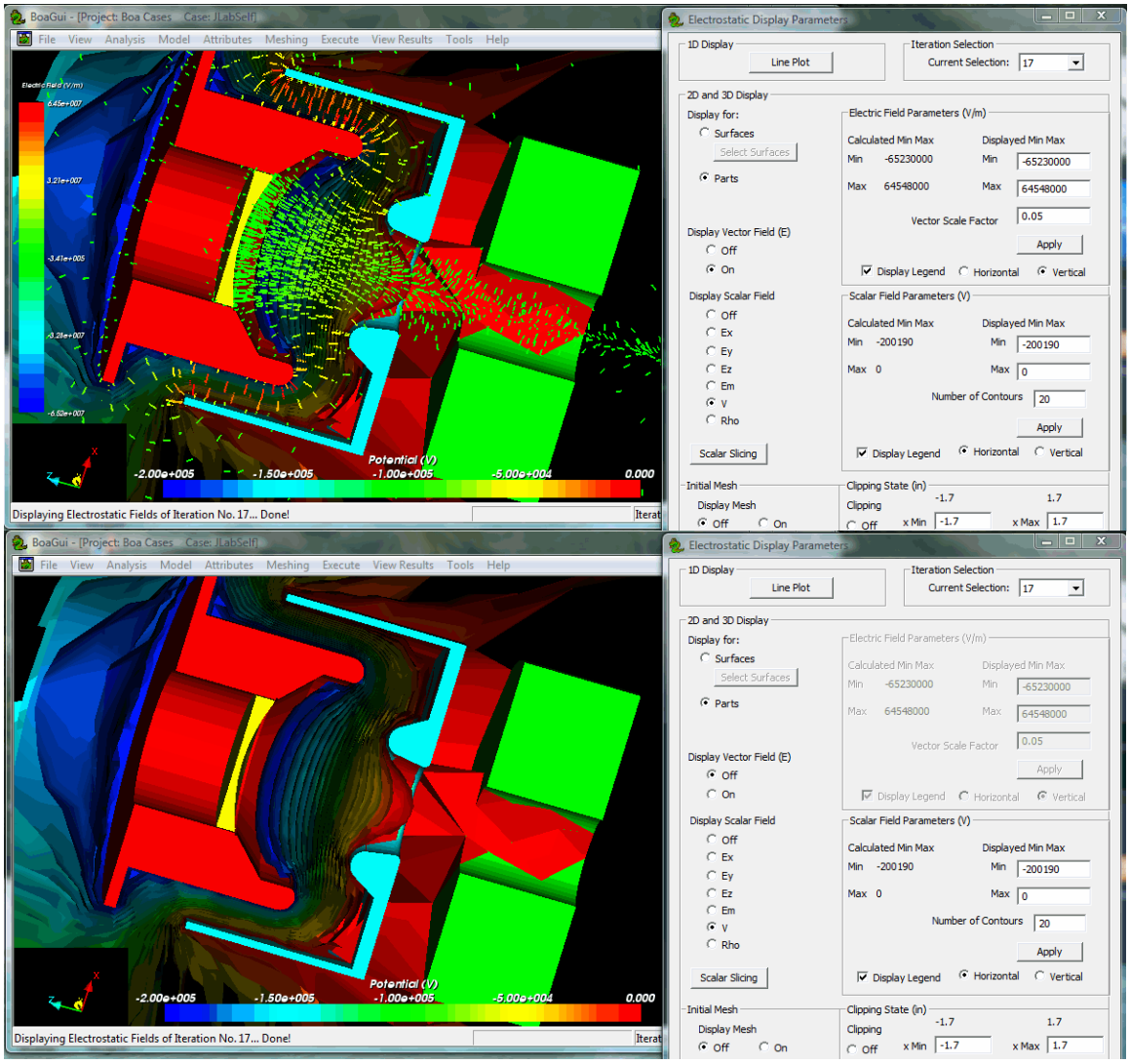

Figure 14 (a) Scalar potential profile with electric field **(b)** Scalar potential only

Both voltage and electric fields can be shown in the same plot as in Fig. 14a, or by singly, as for the voltage in Fig. 14b. One can select a particular particle iteration for field processing from a pull-down list of available iterations, depending on how the user sets the execution parameters. On the control panel, options are available to control the contour limits, number of contour lines, select the field type, display the current mesh, which can be either initial or adapted mesh depending on the selected particle iteration, and the mesh information. It is noted that a vector field type, such as electric field, can be concurrently displayed with another scalar field type, such as charge density. However, two scalar fields such as E_x and E_y cannot be displayed concurrently to avoid visualization confusion.

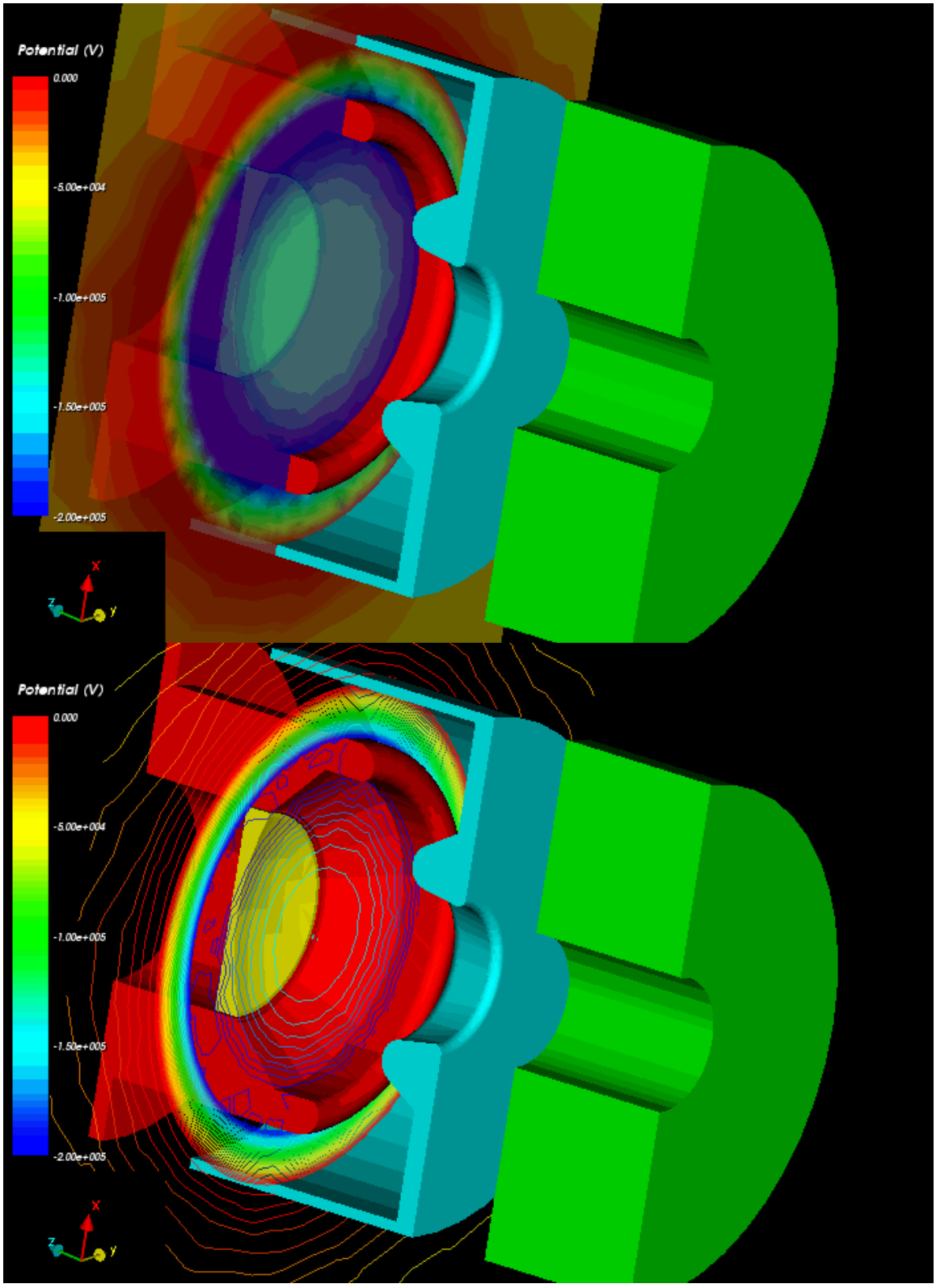

Figure 15 Cross section contour of the scalar potential **(a)** color gradient **(b)** contour lines

When a scalar field is displayed, the *Scalar Slicing* option is available to generate lower dimension contours by slicing a 3D field by an arbitrary plane. Figure 15 shows contouring of the voltage field. The cut plane is normal to the *z* axis and slices through part of the focus electrode. The display of the contours can be either color gradient filled, as in Fig. 15a, or colored lines, in Fig. 15b.

The Line Plot option displays a linear plot of the scalar field along an arbitrary line in space specified by two end points within the boundary of the problem domain. All three components of the electric field along the *z* axis are plotted in Fig. 16. The control panel on the right of the screen allows the user to select the scalar field type, different end points, and number of data points. One can also save the plot for later re-plotting or for export to other plotting packages.

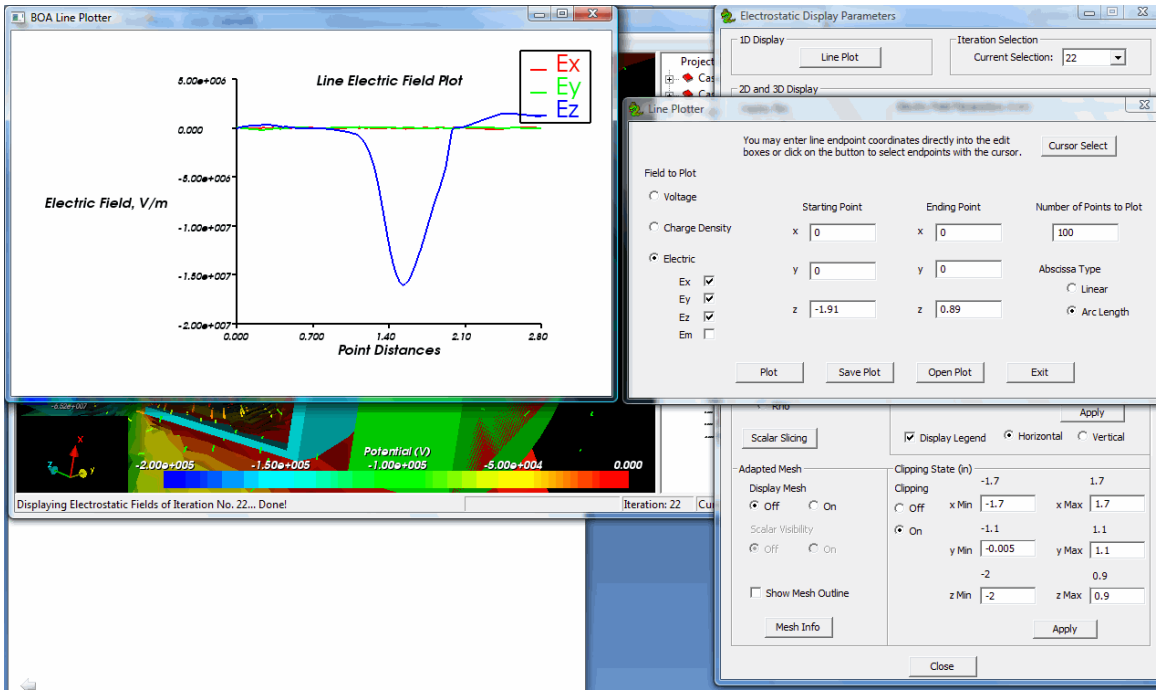

Figure 16 Axial plot of the electric field along the z axis

As mentioned earlier, one can also display the mesh used by the fields for the selected particle iteration. If adaptivity was not enabled, the current mesh would be the same as the initial mesh. The mesh shown in Fig. 17 is actually an adapted mesh showing fine mesh in the particle regions and coarse elsewhere. This mesh has 53,206 tetrahedra and 9,302 vertices. In comparison, the initial mesh, shown earlier in Fig.10, included 104,074 tetrahedra and 17,320 vertices. Consequently, adaptivity reduced the computational resources required by approximately half.

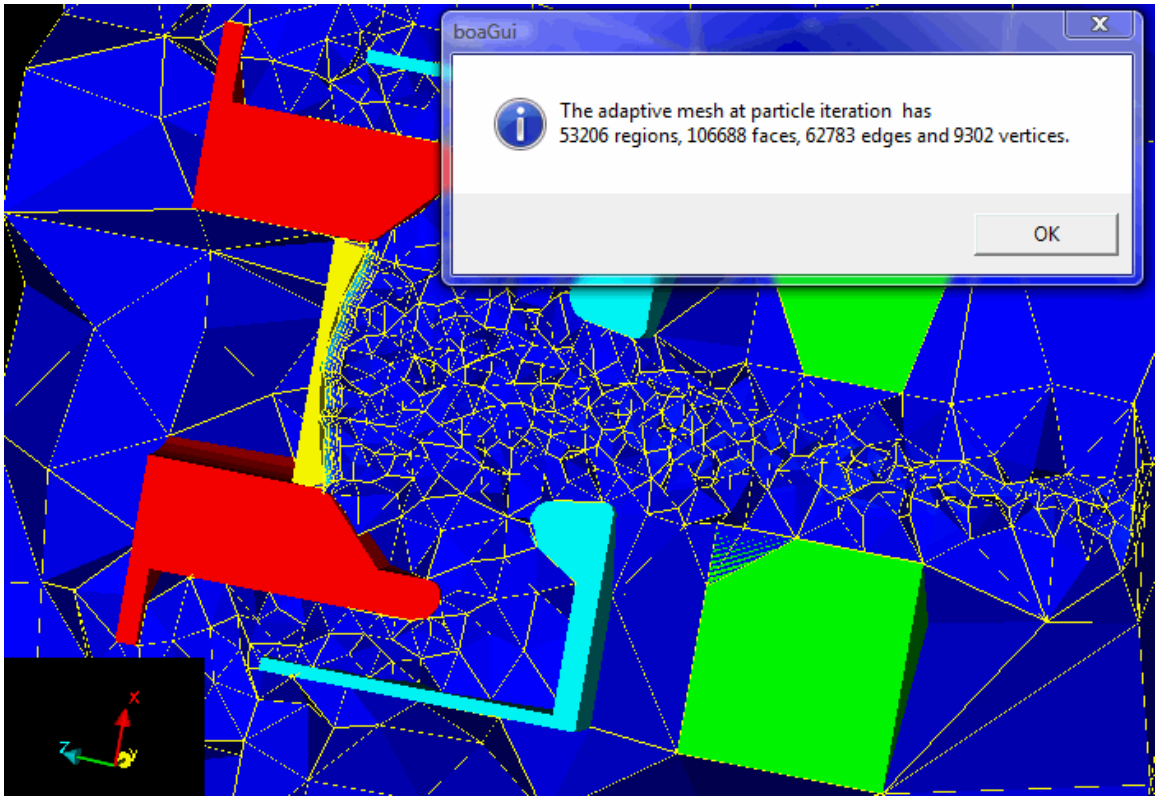

Figure 17 Final adapted mesh shows fine mesh in the particle regions, coarse in the noncritical regions. It has 53206 regions and 9302 vertices

We now continue with the post-processing of the particles from the *View Results* menu to select *Beam Analysis*. In the *Beam Optics Display Control* panel, we can also select one particle iteration from a pull-down list of available iterations, choose to view all or only particles launched from selected emitters, and choose particles of particular generation, if there is secondary emission. We can also select surfaces where particles impact and calculate and display the power density dissipated by these particles.

In the self magnetic field analysis, the electrostatic beam is simulated until convergence, then the converged current is used to compute the self field. The particles with the self field, are pushed again until convergence to generate a second self magnetic field, which is then fixed for the next loop. These steps are repeated as many times as the user specifies in the Execution Parameters window. In Fig. 18a, before the self field becomes active, the particle trajectories are shown, and after two updates of the self magnetic field, the particle trajectories are displayed in Fig. 18b. The beam is clearly focused to a smaller diameter by the self field.

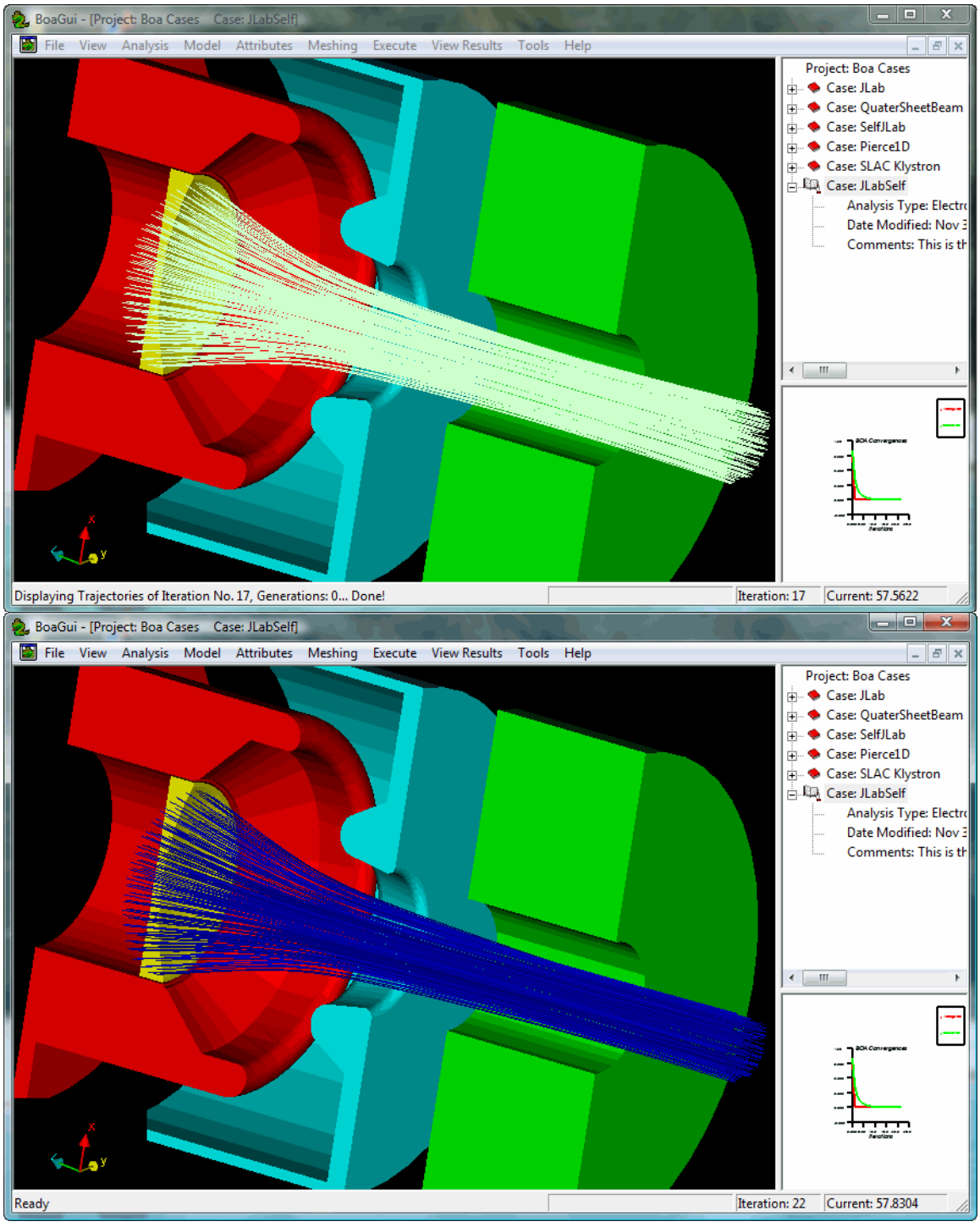

Figure 18 Particle trajectories from a thermionic emitter **(a)** Without self magnetic field **(b)** With self magnetic field, the beam is self focused to a smaller radius

The particle trajectories shown in Fig. 18b are from particle iteration 22. From the *Beam Optics Display Control* panel, which stays on the screen during the post-processing of the particles, one can also display the cross section of the beam perpendicular to one of the major planes (*x*, *y* or *z*). If there is a magnetic field present, particle statistics on this plane

can also be computed and tabulated on the same panel, as shown in Fig.19, and can be saved to a file for later viewing. The particle statistics include kinetic energies, ratios of velocity components, radii and angles with respect to the normalized beam axis and magnetic flux density. In addition, ratios of velocity components, magnetic flux densities and Larmor radii $r_g = c\frac{m}{q}\frac{v_\perp}{B}$ at several prescribed angles can also be tabulated.

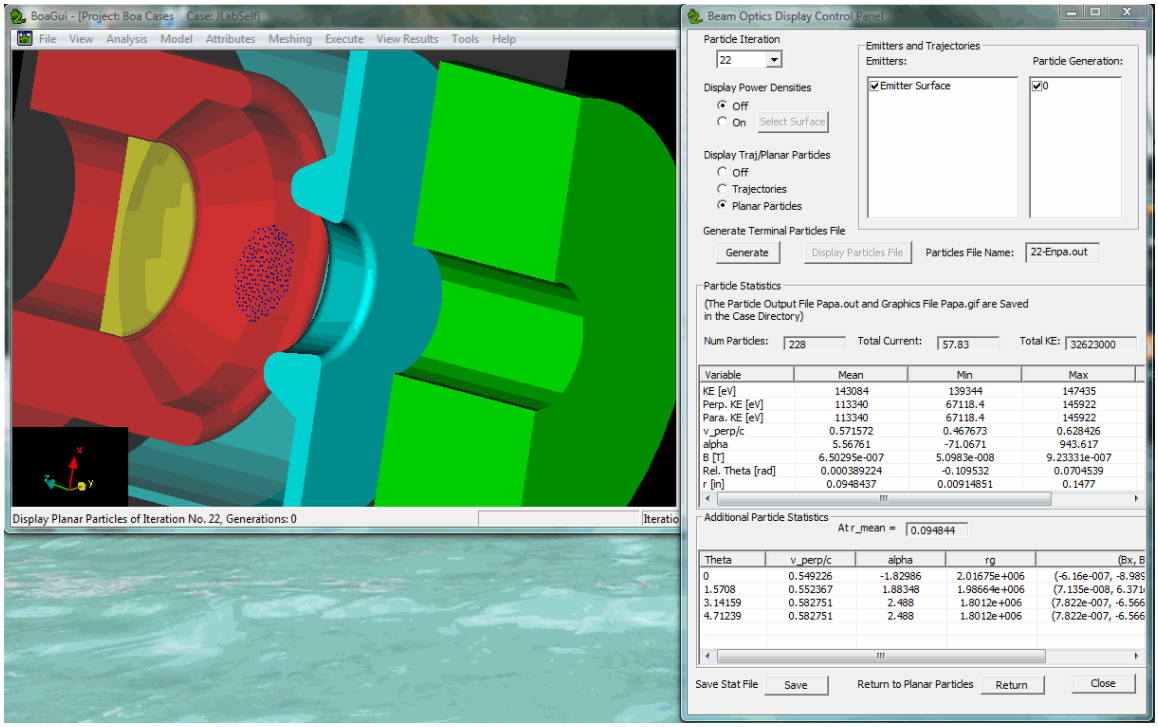

Figure 19 Cross section of the self magnetic beam with the Beam Optics Display Control Panel displaying the particle statistics on this plane

The self magnetic field can be displayed from the *Magnetostatic Display Control* panel. Similar to the *Electrostatic Display Control* panel, the user can select from the pull-down list of available particle iterations which one to post-process for the fields. For self magnetic field analysis, the number of available iterations depends upon the number of self field updates specified in the Execution Parameter window. Vector fields, including magnetic flux density, current density and their components as scalar fields, can be selected for display. One type of vector field and one type of scalar field can be concurrently displayed. However, for visualization clarity displaying either both vector fields or both scalar fields is not permissible.

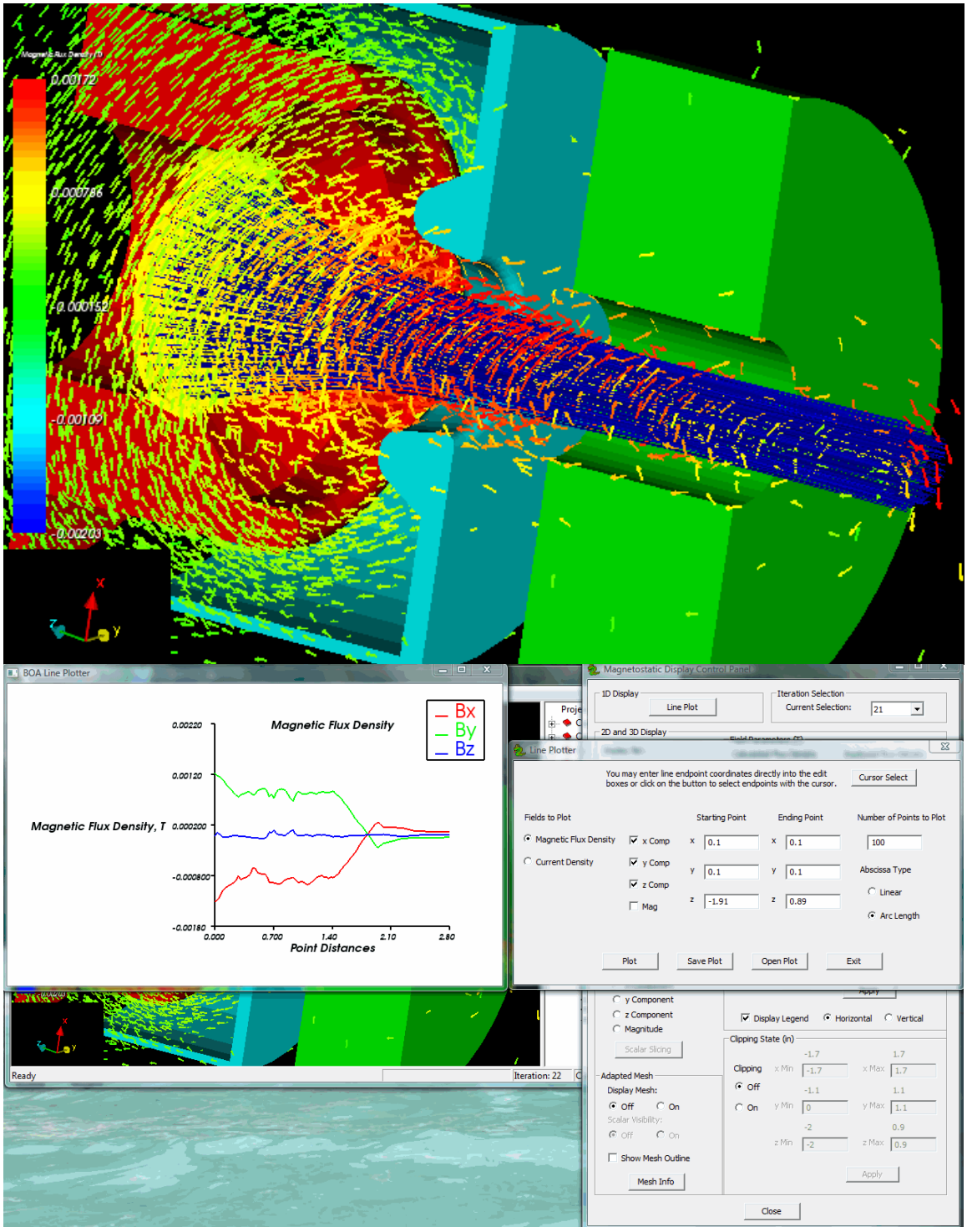

Figure 20 The self magnetic flux density generated by the beam itself (a) In vectors (b) All three components along the *z* axis at *x*=0.1, *y*=0.1

The particle trajectories can also displayed together with the self magnetic field, as demonstrated by Fig. 20a. This figure shows the magnetic flux density vectors circling the beam. All three components of the magnetic flux density along the z axis are also

plotted in Fig. 20b. Again, from the control panel on the right, one can also select the current density to plot and different end point coordinates, save the plot to later re-display, or export for other plotting packages.

Any surface of the model can be prescribed as a secondary emitter. When particles intercept these secondary emitters, they trigger the secondary emission routines in BOA. Depending upon the type of secondary emission, the number of secondary electrons per incidence and the number of secondary generations selected for this emitter, BOA will sample the energies and incident angles of electrons intercepting the emitter to generate a set of scattered electrons to launch back to the particle regions. Let's consider a collector as an example for such analysis. Figure 21 shows this collector with its injecting red disk on the right.

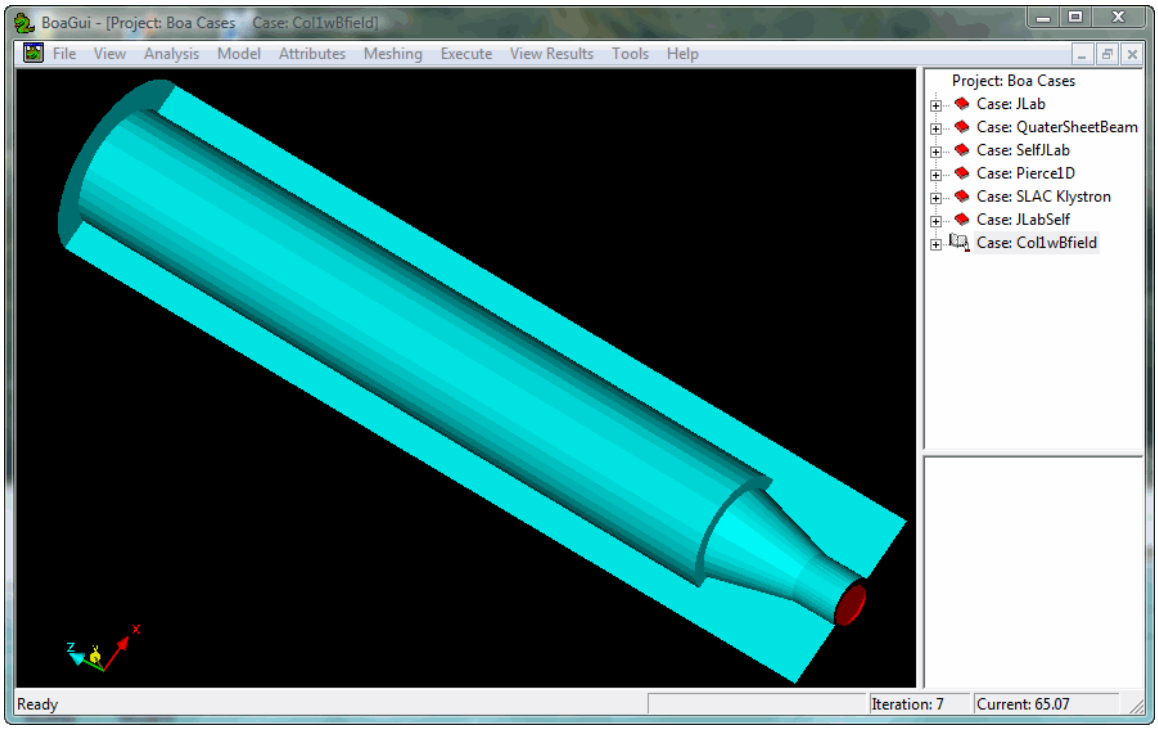

Figure 21 A collector whose interior surface is prescribed as a secondary emitter. Particles will be launched from the red disk on the right

Particles with initial velocities generated elsewhere are launched from this disk. The external magnetic field is imported from Maxwell3D and plotted in Fig. 22 with all three components. Individual component and the magnitude of the magnetic flux density for various end points can also be plotted with the *Linear Plot of Magnetic Flux Density* window.

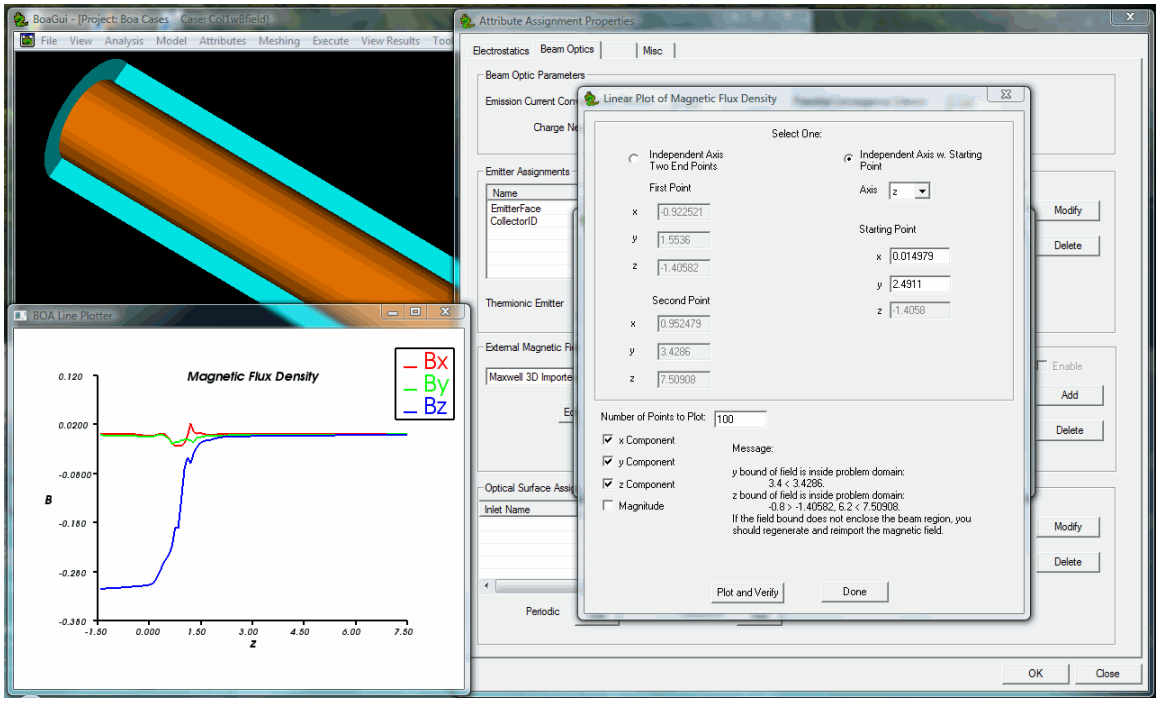

Figure 22 Axial plot of the imported Maxwell3D magnetic flux density, all of its three components. Individual field component including magnitude of the flux and for different end points can also be plotted

The particle trajectories of various generations, including the primary electrons (0^{th} generation), can be displayed from the *Beam Optics Display Control* panel. Figures 23a to 23d show such trajectories. Progressing from only the primary electrons, each plot includes an extra generation of the secondaries. Each generation can have a distinct color set in the *View* menu. From Fig. 23c and 23d, it can be seen that a few electrons of the second and third generation are actually turned back upstream toward the actual emitting source. In practice, such back-streaming electrons must be trapped to avoid possible damage to the emitting source.

Particles on a cross section of the beam are displayed in Figure 24 progressing from (a) only primary electrons to (b) primary and first generation to (c) primary, first and second generation and to (d) primary, first, second and third generation of secondaries.

Again, with the presence of the external magnetic field, particle statistics of those on the cross section can be calculated. They are tabulated and listed on the *Beam Optics Display Control* panel as shown by Fig. 25.

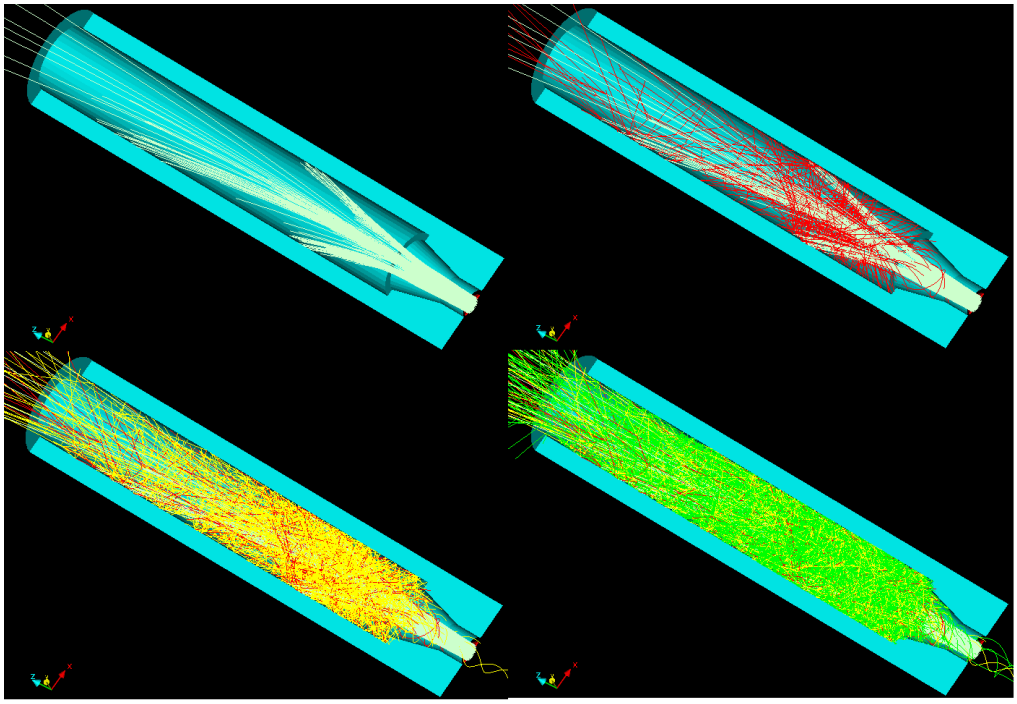

Figure 23 Particle trajectories in a collector: **(a)** Primary electrons **(b)** Primary and first generation of secondaries **(c**) Primary, first and second generation of secondaries **(d)** Primary, first, second and third generation of secondaries

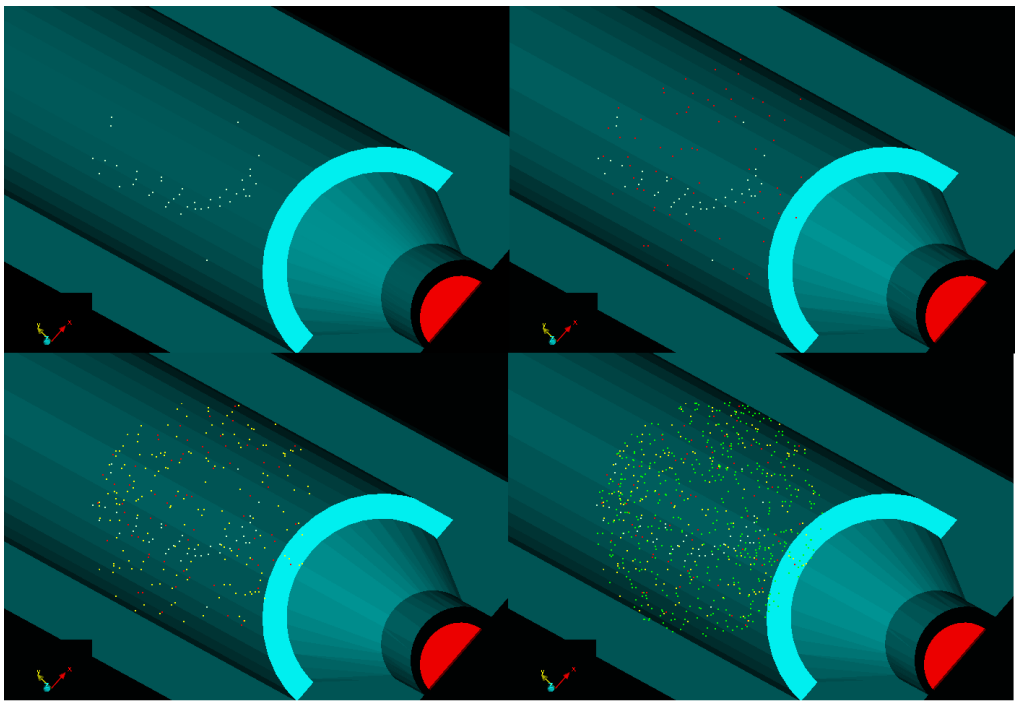

Figure 24 Particles on a cross section plane: **(a)** Primary electrons only **(b)** Primary and first generation of secondaries **(c)** Primary, first and second generation of secondaries **(d)** Primary, first, second and third generation of secondaries

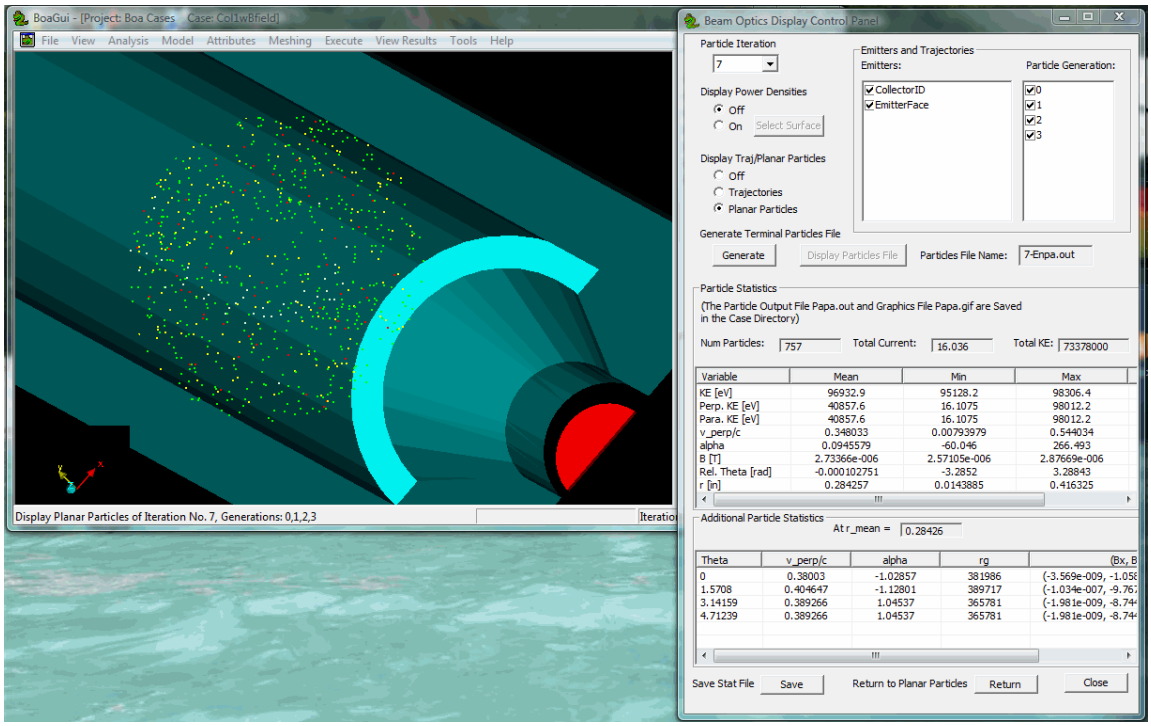

Figure 25 Particle statistics for all generations on the cross section plane

A goal of collector particle simulations is to determine the power dissipated on the interior surfaces of the collector. Such a power density distribution can then be transferred to a thermal analysis code to estimate the temperature profile and thermal stresses. BOA can calculate this power density distribution, and Figure 26 shows the power density caused by all generations of electrons intercepting the interior surface of the collector. The power density is actually very much localized showing a few hot spots. Thus, in the actual design, an improved magnetic field would be required to more effectively distribute the beam to reduce the power density peaks.

Smooth particle loading for quiet start is essential in plasma simulation. We demonstrate the smooth particle loading by BOA in an arbitrary domain. As described earlier, the masking technique is used to load particles., In this technique, the particles are loaded as with the standard PIC schemes in a virtual regular box enclosing problem domain. Only those particles that land on the domain are kept, Figure 27 shows a prism with its unstructured mesh. A cold uniform plasma with initial charge density of $1.0\times10^{-10}\,C/m^3$ is loaded by the inversion of cumulative distribution with uniform probability function and uniform distribution of random numbers. Ions are placed at the same initial locations of electrons as a stationary uniform background. The plasma is cold without any initial velocity. Thus, the particles theoretically should stay motionless, and their

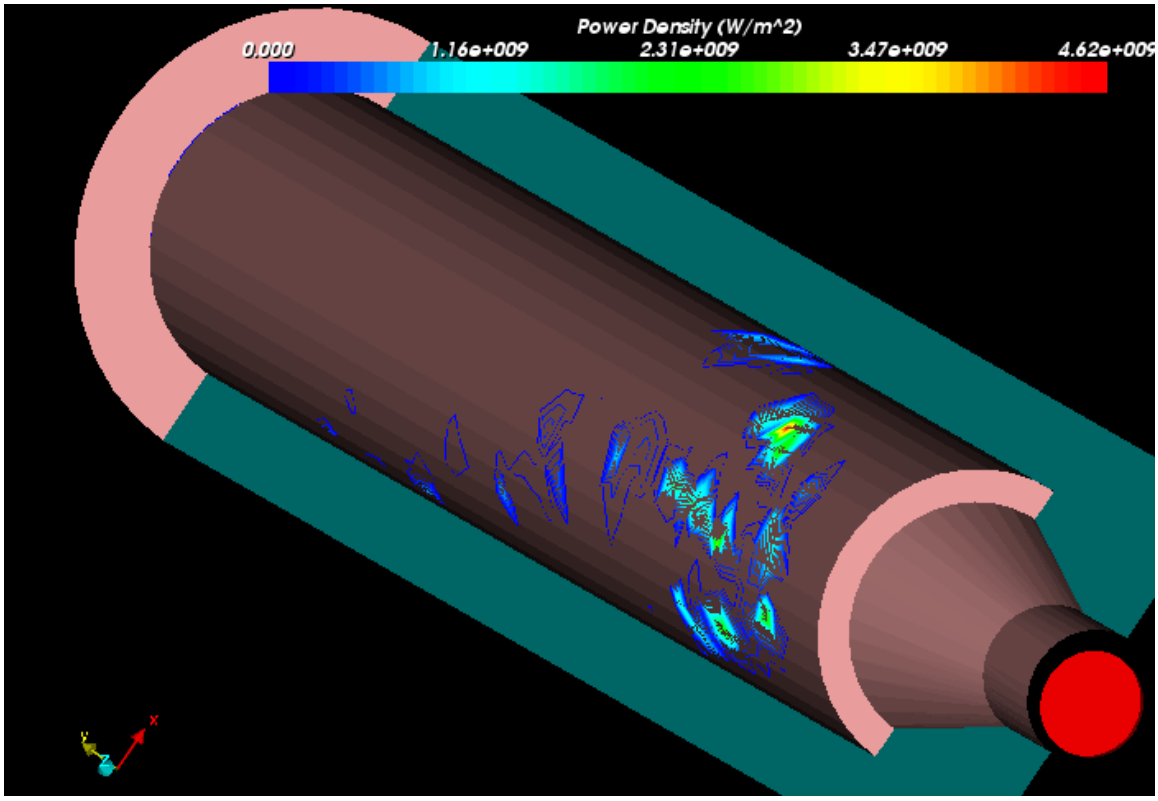

Figure 26 Power density on the interior surface of the collector due to primary and all secondary electrons

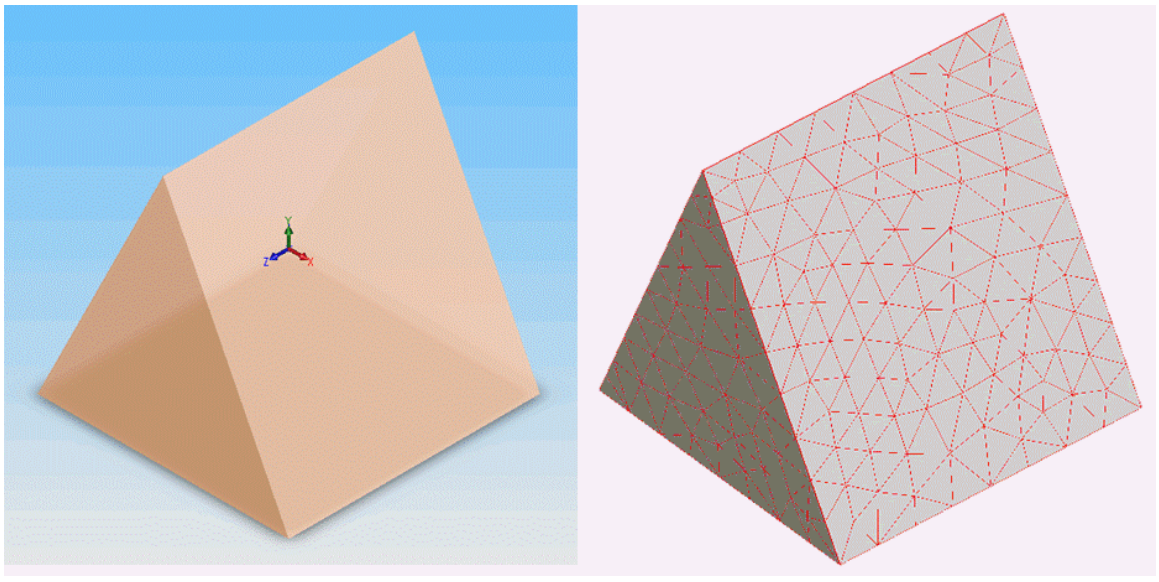

Figure 27 A prism with its mesh: cold plasma is loaded with uniform distribution of number and uniform probability function

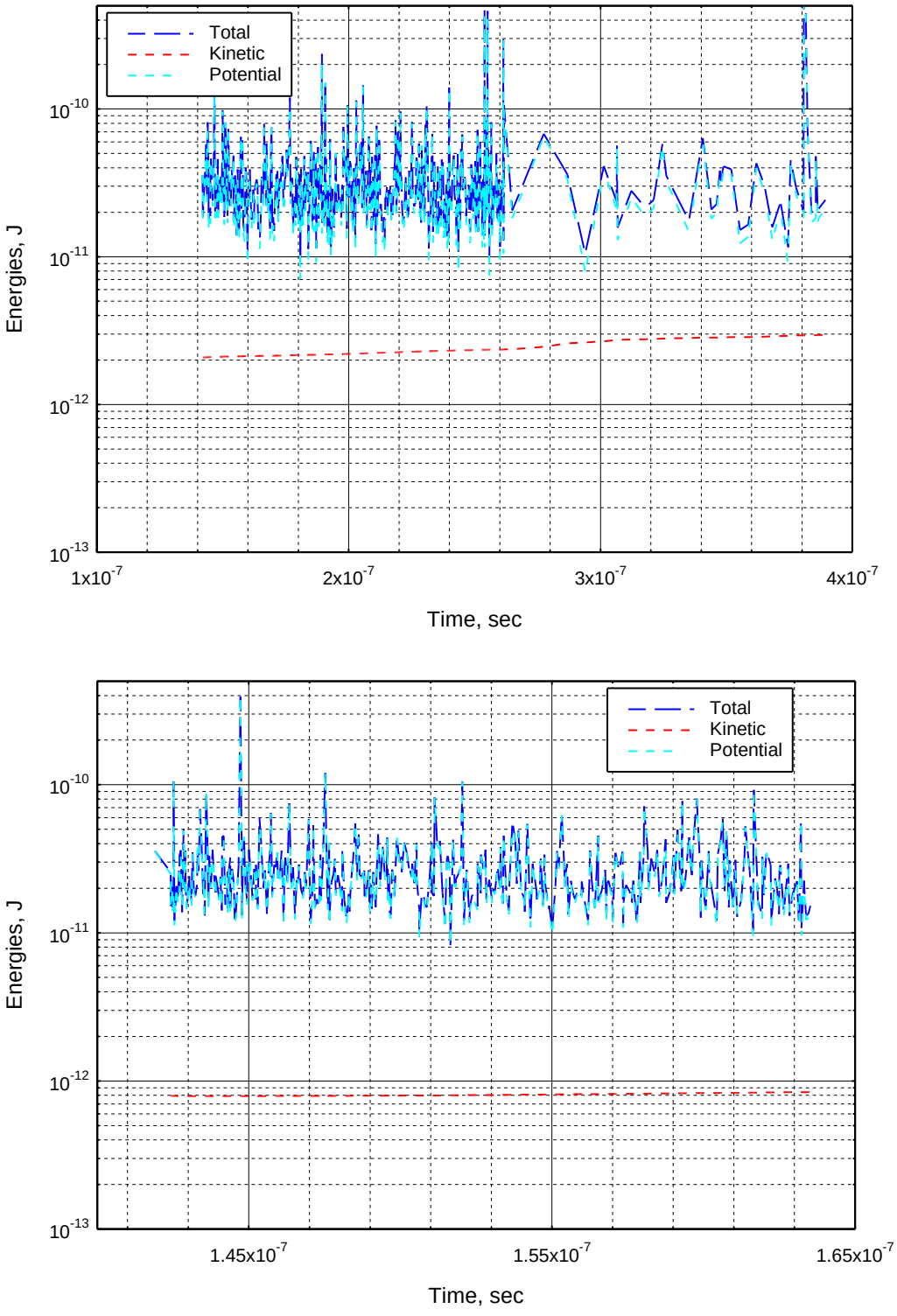

Figure 28 Self heating of cold plasma in a prism: **(a)** No smoothing of electric field **(b)** With electric field smoothing by SPR method provides lower self heating

kinetic and potential energies should be zero and constant through time during the simulation. However, if the loading is not properly done, particularly in a non-orthogonal domain with an unstructured mesh (Figure 27), non-uniformities in the electron-ion distribution can cause field disturbances resulting in unphysical particle motion. With particle loading using uniform probability function with a uniform distribution of random numbers, the self heating of the cold plasma is investigated with the unprocessed and the smoothed electric field by *SPR*. The results are shown in Figure 28b with the combination of smooth loading and smoothed electric field being a better choice. The kinetic energy is lower and stays fairly constant with time.

Unfortunately, it appears that simulating large numbers of particles, i.e., more than 2000, begins to exceed resources for most readily available computers. The computational cost of tracking more particles in an adapting, unstructured mesh begins requiring excessive memory for data storage. This probably makes this approach currently unfeasible for plasma simulations of interest where upwards of 2 million particles are required. A more efficient technique would be to track the particles on a structures mesh while the electromagnetic fields are solved on an adaptive, unstructured mesh. This would provide high accuracy for both the field and particle simulations, and efficient routines are available for transferring information between the two meshes. CCR is currently pursuing this approach.

Summary

BOA is a particle-in-cell code for analysis and design of electron beams and collectors, and simulation of plasmas in complex 3D geometries. It has fully automatic and adaptive meshing, allowing quick problem setup and reduction of computer resources. Its graphical user interface is simple, intuitive and allows import of geometry from commercial CAD packages. Its post-processing is powerful with region, surface and line plotting of the fields, particle trajectory display with particle statistics and power density.

We are continuing to improve and add features to the code. The immediate plan is to parallelize BOA using OpenMP for shared memory multiprocessor architecture, Such parallel systems are becoming more and more widespread and affordable. We are also implementing electron gun optimization algorithms using prescribed cost functions. This will dramatically reduce the cost of electron gun design while simultaneously improving the performance.

The program was successful in implementing thermal electrons, secondary emissions, and self magnetic field calculations. The BOA GUI was also upgraded significantly, and CCR is receiving interest from the microwave tube and semiconductor equipment industry for the code. Implementation of PIC analysis was partially successful. Computational resource requirements for modeling more than 2000 particles begin to

exceed the capability of most readily available computers. Modern plasma analysis typically requires modeling of approximately 2 million particles or more. The problem is that tracking many particles in an unstructured mesh that is adapting becomes inefficient. In particular memory requirements become excessive. This probably makes particle tracking in unstructured meshes currently unfeasible with commonly available computer resources. Consequently, Calabazas Creek Research, Inc. is exploring hybrid codes where the electromagnetic fields are solved on the unstructured, adaptive mesh while particles are tracked on a fixed mesh. Efficient interpolation routines should be able to transfer information between nodes of the two meshes. If successfully developed, this could provide high accuracy and reasonable computational efficiency.

References

1. W. B. Mori, *Recent Advances and Some Results in Plasma-Based Accelerator Modeling, Advanced Accelerator Concepts*, Tenth Workshop, Edited by C. E. Clayton and P. Muggli, American Institute of Physics, 2002.

2. J. P. Verboncoeur, *Particle-in-Cell Techniques*, Short Course given at U.S. Army Benet Laboratories in March 2002; also given at the IEEE International Conf. On Plasma Science Minicourse in 2002 in Banff Canada, and in 2000 in Monterey CA.

3. C. K. Birdsall and A. B. Langdon, *Plasma Physics via Computer Simulation*, McGraw-Hill Book Company (New York), 1985.

4. T. Bui et. al., *Beam Optics Analysis – A 3D Finite Element Charged Particle Code with Adaptive Meshing*, Paper 9.3, IVEC, Monterey, CA, USA, 2002.

5. T. Bui et. al., *Initial Operation of Beam Optics Analysis, a 3D, Charged Particle Code with Adaptive Meshing, Paper 2A02*, ICOPS, Banff, Alberta, Canada, 2002.

6. W. H. Press et. al., Numerical Recipes in C (2^{nd} ed.), Cambridge University Press (Cambridge, U.K.), 1992.

7. T. Hughes, *The Finite Element Method*, Prentice-Hall (Englewood Cliffs, NJ), 1987.

8. P. P. Silvester and R. L. Ferrari, *Finite Elements for Electrical Engineers* (3^{rd} ed.), Cambridge University Press (Cambridge, U.K.), 1996.

9. G. Strang and G. Fix, *An Analysis of the Finite Element Method*, Prentice-Hall (Englewood Cliffs, NJ), 1973.

10. B. Szabo and I. Babuska, *Finite Element Analysis*, John Wiley & Sons (New York), 1991.

11. O.C. Zienkiewicz and J.Z. Zhu, *The superconvergent patch recovery and a posteriori error estimates. Part 1: The recovery technique*. International Journal of Numerical Methods in Engineering, 33:1331-1364, 1992.

12. O.C. Zienkiewicz and J.Z. Zhu, *The superconvergent patch recovery and a posteriori error estimates. Part 2: The recovery technique*. International Journal of Numerical Methods in Engineering, 33:1365-1382, 1992.

13. N. Yan and A. Zhou, *Gradient recovery type of a posteriori error estimates for finite element approximation on irregular meshes*, Computer Methods in Applied Mechanics and Engineering, Vol. 190, 2001.

14. M. Ainsworth and J.T. Oden, *A Posteriori Error Estimation in Finite Element Analysis*, John Wiley & Sons (New York), 2000.

15. O.C. Zienkiewicz and R.L. Taylor, *The Finite Element Method, Vol. 1* (4^{th} ed.), McGraw-Hill (London, U.K.), 1989.

16. L.B. Wahlbin, *Superconvergence in Galerkin Finite Element Methods*, Vol. 1605 of Lecture Notes in Mathematics. Springer-Verlag, 1995.

17. J. P. Boris, *Relativistic Plasma Simulation – Optimization of a Hybrid Code, Proc. Fourth Conf. Numerical Simulation of Plasmas*, p. 3-67, (Washington DC), 1970.

18. K. L. Cartwright, J. P. Verboncoeur and C. K. Birdsall, *Loading and Injection of Maxwellian Distributions in Particle Simulations*, J. Comp. Phys. 162, 483-513, 2000.

19. J. H. Ferziger, Numerical Methods for Engineering Application, John Wiley & Sons (New York), 1981.

20. J. P. Verboncoeur, A. B. Langdon, and N. T. Gladd, *An Object-Oriented Electromagnetic PIC Code*, Comput. Phys. Comm. 87, 199-219, 1995.

21. J. P. Verboncoeur, *Symmetric spline weighting for charge and current density in particle simulation*, J. Comp. Phys. 174, 421-427, 2001.

22. T. Bui et. al., *Code Development of a 3D Finite Element Particle-In-Cell Code with Adaptive Meshing*, IVEC, Noordwijk, The Netherlands, 2005.

23. T. Bui et. al., *Code Development of a 3D Finite Element Particle-In-Cell Code with Adaptive Meshing*, ICOPS, Monterey, CA, USA, 2005.

24. A. Bossavit, *Computational Electromagnetism*, Academic Press (San Diego CA, Chestnut Hill MA), 1998.

25. M. Barton et al., *New vector finite element for three-dimensional magnetic field computation*, J. Appl. Physics **61**(8), 15 April 1987.

26. H. Igarashi, *On the Property of the Curl-Curl Matrix in Finite Element Analysis With Edge Elements*, IEEE Transactions on Magnetics, Vol. 37, No. 5, September 2001.

27. J. Manges et al., *A Generalized Tree-Cotree Gauge for Magnetic Field Computation*, IEEE Transactions on Magnetics, Vol. 31, No. 3, May 1995.

28. Y. Liu et al., *Eddy-Current Computations Using Adaptive Grids and Edge Elements*, IEEE Transactions on Magnetics, Vol. 38, No. 2, March 2002.

29. Z. Ren, *Influence of the R.H.S. on the Convergence Behaviour of the Curl-Curl Equation*, IEEE Transactions on Magnetics, Vol. 32, No. 3, May 1996.

30. G. Rodrigue and D. White, *A vector finite element time-domain method for solving Maxwell's equations on unstructured hexahedral grids*, SIAM Journal of Scientific Computation Vol. 23, No. 3 p. 663-706, 2001.

31. J. P. Verboncoeur, *Particle simulation of plasmas: review and advances*, Plasma Physics and Controlled Fusion, Vol. 47, A231-A260 (2005).

32. J. Jin, *The Finite Element Method in Electromagnetics*, Second Edition, John Wiley & Sons (New York), 2002.

33. C.T. Kelley, *Solving Nonlinear Equations with Newton's method*, SIAM (Philadelphia), 2003.

34. C.T. Kelly, *Iterative Methods for Linear and Nonlinear Equations*, SIAM (Philadelphia), 1995.

35. J.P. Verboncoeur, *Particle simulation of plasma: review and advances*, Plasma Physics and Controlled Fusion, Vol. 47, A231-A260 (2005).

36. T. Bui, *Magnetostatic Solver – Vector Finite Element for 3D Magnetostatic Problems*, CCR Internal Design Document, 2002.

37. J.F. Remacle, M.S. Shephard, *An Algorithm Oriented Mesh Database*, International Journal for Numerical Methods in Engineering, 2002.

38. J.F. Remacle, J.E. Flaherty, M.S. Shephard, *A Parallel Algorithm Oriented Mesh Databas*, Engineering With Computers, 2002.

39. M.S. Shephard and M. Georges, *Automatic three-dimensional mesh generation by the finite octree technique*, Intl. J. Numer. Meths. Engrg., 32 p. 709-749, 1991.

40. J.E. Flaherty, R. Loy, M.S. Shephard, B. Szymanski, J. Teresco and L. Ziantz, *Adaptive local refinement with octree load-balancing for the parallel solution of three-dimensional conservation laws*, J. Parallel Distrib. Comput., 47 p. 139-152, 1998.

41. J. O'Rourke, *Computational Geometry in C* (2nd ed), Cambridge University Press (Cambridge), 1998.

42. R.K. Morley, P. Shirley, K. Morley, *Realistic Ray Tracing* (2nd ed), AK Peters Ltd, 2003.

43. D.F. Kapraun, *Monte Carlo Techniques for Predicting Electron Backscattering*, Master Thesis, North Carolina State University, 2001.